MODELING INFORMATION QUALITY EXPECTATION IN UNMANNED AERIAL VEHICLE SWARM SENSOR DATABASES

THESIS

Patrick D. Baldwin
First Lieutenant, USAF

AFIT/GCS/ENG/05-01

DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GCS/ENG/05-01

MODELING INFORMATION QUALITY EXPECTATION IN UNMANNED AERIAL VEHICLE SWARM SENSOR DATABASES

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science

Patrick D. Baldwin, B.S.

First Lieutenant, USAF

March, 2005

AFIT/GCS/ENG/05-01

MODELING INFORMATION QUALITY EXPECTATION IN UNMANNED AERIAL VEHICLE SWARM SENSOR DATABASES

THESIS

Patrick D. Baldwin, B.S.

First Lieutenant, USAF

Approved:

/signed/	
Lt Col Michael L. Talbert Thesis Advisor	Date
/signed/	
Maj Christopher B. Mayer Committee Member	Date
/signed/	
Dr. Gilbert L. Peterson Committee Member	Date

Dedication

Our lives are shaped by those people who dedicate a part of themselves to us. Francis Dillon Baldwin was a man who lived a peaceful, honorable life dedicated to his family. He was also my Father. He worked long hours to ensure we had a roof over our head and three square meals on our plates every day. He was our champion and was proud of the way we lived our lives. The day I left for basic training, he told me that he couldn't be more proud of my decision to join the Air Force. Throughout my career he has been there to hear my excited calls when good things happened and to listen when disappointments came my way. In both cases he always knew the right words to say.

When I called him to tell him of my selection to AFIT, he marveled at the opportunities the Air Force had afforded me and he was always interested to hear how the next homework assignment, test, paper or grade report came out.

On October 4, 2004, during my thesis quarter, my Father passed away from complications of a long illness. My family and I miss him, but we know he's looking over us.

This work is dedicated in his memory.

I Love You Dad

Acknowledgements

AFIT is not a mission anyone undertakes on his or her own. Several people contribute in many ways to the successful completion of a degree at this institution.

First and foremost I want to thank my family for their unending support of me, my career and my time at AFIT. To my beautiful wife who has pushed me to yet another height in my life with her ability to be a wife, mother, teacher, party planner, assistant coach, taxi driver, spiritual leader and confidant all with little help from me during the past 18 months. And to my three awesome kids for greeting me at the door every night with open arms and hearts and for understanding as best they could that Daddy was doing this for them as well as himself. I'm done now you guys, lets play.

To my advisor, LtCol Michael Talbert I extend my heartfelt gratitude for your guidance, mentorship and friendship during this time. Thank you for your ability to guide my research with an open mind and allowing me to work a small piece of your research puzzle.

I would like to thank my sponsor, Dr. Douglas Holzhauer from the Air Force Research Laboratory in Rome, NY for his support of this effort. Also from AFRL, I would like to thank Jason Moore, Chad Salisbury, and Patrick Fisher for their assistance with the Audit Trail Viewer visualization software.

No one gets through this place without a little help from their friends. Charlie P, Kevin B, Kevin O, and Matt R, you guys are the best. Your friendship and support was invaluable and I never would have made it through without you guys. I wish you the best of luck in everything you do and I hope our paths cross again.

Finally I want to thank God for providing me with the wisdom, patience and abilities to make it through another challenge in my life.

Patrick D. Baldwin

Table of Contents

Page

Page

List of Figures

List of Tables

List of Abbreviations

Abbreviation		Page
AWACS	Airborne Warning and Control System	2
JSTARS	Joint Surveillance Target Attack Radar System	2
SAR	Synthetic Aperture Radar	2
ABM	Air Battle Managers	2
UAV	Unmanned Aerial Vehicle	2
ISR	Intelligence, Surveillance and Reconnaissance	2
WLAN	Wireless Local Area Network	3
IR	Infrared	4
GCCS	Global Command and Control System	7
COMMINT	Communications Intelligence	10
EO	Electro optical	12
RF	Radio Frequency	16
UTC	Universal Time Coordinated	20
MSL	Mean Sea Level	20
GUI	Graphical User Interface	21
GPS	Global Positioning System	31
SPAT	Swarm Position Analysis Tool	42
API	Application Programming Interface	58
ATV	Audit Trail Viewer	58
AFRL/IF	Air Force Research Laboratory Information Directorate	61
FOV	Field Of View	75
DTED	Digital Terrain and Elevation Data	79

List of Algorithms

AFIT/GCS/ENG/05-01

Abstract

Swarming Unmanned Aerial Vehicles (UAVs) are the future of Intelligence, Surveillance and Reconnaissance (ISR). Swarms of hundreds of these vehicles, each equipped with multiple sensors, will one day fill the skies over hostile areas. As the sensors collect hundreds of gigabytes of data, telemetry data links will be unable to transmit the complete data picture to the ground in real time. The collected data will be stored on board the UAVs and selectively downloaded through queries issued from analysts on the ground.

Analysts expect to find relevant sensor data within the collection of acquired sensor data. This expectation is not a quantified value, rather a confidence that this relevant data exists. An expectation of the likely quality of the available sensor information is determined by the user through the use of the methods and tools developed in this thesis.

This work develops swarm coverage analysis models using position in time data from the swarm. With these models, a geometric analysis of the swarm is conducted that shows analysts when and where the swarm likely collected sensor data most relevant to a need. Convex hulls are used to calculate areas of coverage as well as swarm and sensor densities. Target profiling algorithms are developed that show target coverage over time from the swarm for each sensor type. Target-centric and sensor-centric analyses allow analysts to quickly determine where individual swarm agents were relative to a target at any point during the mission. Finally a series of visualizations of the swarm and targets are created that allow the analyst to view swarm activity from the perspective of individual swarm members or targets.

MODELING INFORMATION QUALITY EXPECTATION IN UNMANNED AERIAL VEHICLE SWARM SENSOR DATABASES

1. Introduction

“Fog of war” refers to the inability of war planners, command and control entities and front line warriors to get a clear picture of battlefield activity and status because of too much, too little, inaccurate or untimely information. For centuries, the acquisition and dissemination of information for war fighting purposes has been pivotal in both victory and defeat. Communication between entities gathering information and entities needing information has been key in war throughout time. The physical communication methods, from carrier pigeon to encrypted satellite communication link, have evolved and improved at an incredible rate. Today the military has unprecedented information gathering capabilities. From special forces foot soldiers to high altitude reconnaissance satellites, our ability to see the activities of our enemies gives the United States a tremendous advantage in combat. Somewhere between the foot soldiers and the satellites are flying machines that are used on a constant basis to peer into enemy territory. Unmanned Aerial Vehicles are the future of intelligence gathering operations. With the advent of swarming technology, and the ability to place multiple sensors on board a single vehicle, information can be gathered significantly faster than the ability to download the information to the user. This work introduces methods of analyzing a swarm of unmanned aerial vehicles and determining, through their movements over a target field, when sensor data was likely to be gathered on specific targets.

1.1 Background

In the 1950's and 1960's the U-2 Dragonlady, SR-71 Blackbird and RC-135 Rivet Joint aircraft were the primary intelligence gathering tools of the United States Air Force. Their ability to fly high and/or fast protected their human cargo while gathering intelligence with high resolution cameras, high powered zoom lenses and sophisticated electronic surveillance equipment. In the late 1970's the E-3 Airborne Warning and Control System (AWACS) planes were introduced. These aircraft act as mobile air traffic control centers, scanning the skies and seas for enemy aircraft and maritime vessels and controlling the air battle with deadly efficiency. In the 1990's, the United States acquired such tools as the E-8 Joint Surveillance Target Attack Radar System (JSTARS). Through the use of Synthetic Aperture Radar (SAR), JSTARS is capable of realtime acquisition of moving targets on the ground. Human air battle managers (ABM) on board communicate directly with controllers managing the forces engaged in ground combat. The primary disadvantages of these systems are their size, lack of stealth, on-board human operators and inability to get close enough to find small targets without risking those on board.

The Department of Defense is currently undergoing a transformation from manned to unmanned aerial vehicles (UAVs) for their intelligence, surveillance and reconnaissance (ISR) missions. With their ability to fly deep into enemy territory virtually undetected, loiter for hours at a time and collect myriad types of information without putting a human life in danger, these UAVs are rapidly becoming the preferred method for gathering intelligence on the enemy as well as tracking allied forces. Operations ENDURING FREEDOM and IRAQI FREEDOM are but the latest in the recent applications of UAV technology to survey the enemy. United States Air Force Predator UAVs are currently seeking out and actually destroying enemy targets using a combination of visual sensors and on-board weapon systems. The RQ-4 Global Hawk UAV is a long range high altitude reconnaissance bird that is capable of sustained flights of 36 hours or more, scanning some 40,000 square miles of terrain during a single mission.

As these and other smaller UAVs become equipped with multiple types of data gathering sensors, the ability to view the data from all of the sensors in real time is greatly diminished. Compounding the challenge is the advent of swarming technology which allows UAVs to fly in groups of 1,000 or more. Efficient methods are needed for determining what data is relevant and for retrieving only that data.

Recent research [7] [9] has focused on the ability of UAVs to perform their missions *en mass* using swarming technology. While these works are simulation studies only, they lay the groundwork for improved future operational capabilities. Swarming allows several UAVs (henceforth referred to as agents) to perform missions as a single unit. During these missions, the swarm of agents is flown as a cohesive unit over a predetermined geographic area in search of targets of interest. Control of the agents is handled by algorithms that dictate where the individual agents are with respect to each other at any given time. Rather than determine the exact flight path of each agent, a set of rules is laid out that form the behavior of the agents within the swarm. These rules are responsible for adjusting the heading, speed and orientation of the agents with respect to each other based on interaction between agents.

War planners are constantly gathering information needed to make informed decisions about the status of virtually everything on the battlefield. Hundreds of assets are working together in an enormous space gathering data about the current happenings and feeding that information to each other and to the decision makers on the ground. UAVs are fitted with any number of sensors that look, listen, feel and monitor several different phenomena. Data collected by these sensors needs to be processed and stored for retrieval sometime in the future. In some cases, the data is stored either on the UAV that collected the information or transmitted over a wireless local area network (WLAN) to a “mother” ship acting as the storage facility for the swarm. Because UAVs can fly where other assets may not, data from them can be very valuable to the planners. A key challenge facing the war planners and intelligence analysts is retrieving this data in an efficient manner.

1.2 Problem Description

Imagine a swarm of 100 UAVs sent on a mission to survey the mountains of Afghanistan. Of these, all 100 have video sensors, 50 have Infrared (IR) sensors and 30 have SAR capability. To complicate matters, some of the agents have all three sensors on board. As the swarm moves, these 180 sensors are all collecting data about their surroundings. With the quantity of data being collected, it is difficult to transmit every bit of data to the ground using current information transmission methods. Simply put, the satellite data links do not have the transmission capability to move that quantity of information in real time. Furthermore, it is extremely difficult for the analysts to sift through such a large amount of data to find the possibly small percentage that can give needed information about specific targets.

Data collected by the sensors is stored in a database on board the UAV. This database receives queries from analysts needing the collected information. As the queries are processed, an application retrieves the information from the database, bundles it up in data packet(s) and sends it to the requesting agent via a data transmission link of limited bandwidth. If too many queries are sent to the database, or if the queries return too much information, these data links become overburdened and are unable to handle the requests in a timely manner. The net result is a high percentage of unneeded information being sent that needs to be scoured for valuable information

Information quality expectation deals with predicting, through analysis of swarm position in time data, the most advantageous times to ask for information from the UAV swarm sensor database. By reducing the requested information to only the relevant pieces, the quantity of information sent through the data links is reduced.

Furthermore, by looking at data from sensors of the same type, it becomes possible to corroborate readings from these sensors, increasing the confidence that the information is correct. The problem addressed in this research is the vast amounts of data collected by sensors on board the UAVs. With huge amounts of information being collected, methods are needed that can identify the sensors that have collected relevant data and the times when this collection occurs. The ability to do this without having to download all of the information and sift through it saves time and bandwidth.

1.3 Research Focus

This research focuses on developing methods and algorithms for assessing the degree of coverage of UAV swarm elements over a set of distinct target points in an area of interest. Metrics such as swarm and sensor density, position, bearing, and coverage over time are calculated by analyzing the movement of the swarm over the target field. This helps determine what information can be efficiently obtained from swarming, multi-sensor laden UAVs with a high degree of certainty (high expectation) that the information meets the time-space needs of the user. The thrust of this research is the development of methods that equip operators and war planners with efficient tools to rapidly access the *relevant* information stored in the sensor databases.

Relevance is defined as a "differential subset of information that makes it of interest for the receiver" [13]. This is a key concept, because relevance is determined from the perspective of the receiver of the data in a time-space information need context. The acquisition of only relevant data is important to war planners and analysts because their decision making ability is tied to the information that is received through all channels. In this case, relevance refers to the degree to which a designated piece of information accurately portrays specific events in the area of interest that are of a significant nature. As an example, an hour of video from a

UAV may have 30 seconds of coverage of a column of tanks moving toward friendly positions. A great deal of the video is of little anticipated use to the analysts, but the 30 seconds of tank video is extremely relevant to the commanders in charge of the friendly positions.

Several research fields are encompassed including UAV swarm technology, sensor technology, information storage and retrieval, and networks. By drawing on technologies from each of these fields, the methods developed in this work allow the individuals analyzing the data to better decide what information to ask for from the sensor database. The data requests are made with a high degree of confidence that the data returned is able to be used as a point of reference for making critical decisions.

1.4 Assumptions

Presently, The field of UAV swarms exists largely in the theoretical world. Because of this, some assumptions need to be made about the capabilities of UAV swarms as operational entities. Sensor databases are relatively new and have not been implemented in UAV swarms except in models. In the interest of keeping the scope of this research to the information expectation domain, and not trying to delve into the UAV swarming discipline or sensor database arena, assumptions are made about both research areas.

1.4.1 UAV Swarms. Because UAV swarms are autonomous, the swarm is viewed as being flown as a single entity. The operator of the swarm dictates a mission profile and it is up to the controlling algorithms to determine where each of the agents is at any given time. It is reasonable to assume that actual or estimated tracking data is maintained on the swarm entities or recorded in self-reports within each agent. Without it, an errant agent may not be detected until it had strayed so far from the swarm as to be beyond the reach of the swarm controllers and thus unrecoverable. This being the case, it is also reasonable to assume that this data is maintained in real time and is accessible to the operator through the Global Command and Control System (GCCS) or a like data repository network. For the purpose of this research it will be assumed that there is access in near real time to this tracking data.

1.4.2 Sensor Database. Sensors are manufactured by dozens of companies and are capable of being identified by and providing output with literally hundreds of different parameters. The sensor database for this work consists of very basic information. *Information quality expectation* as presented in this work is based primarily on swarm geometry and proximal data. Because of this, the database contains the tracking data and calculated data that is derived from the tracking data only. Modeling the sensor data per an area for future work and as such the extent to which the sensors are modeled in this work consists of simply defining the UAVS as having particular types of sensors.

2. Previous Work

Over the past 15 years, the concept of using swarming technology in the military world has taken hold and become a major research thrust. By harnessing the capabilities of today's UAVs and combining swarming technology and rapidly-evolving sensor technology into an integrated package, the United States military stands to gain a tremendous advantage in battlefield ISR. Coupled with this technology is the ability to store and retrieve the volumes of gathered information. The work accomplished by hundreds of individuals and teams previously sets the groundwork for the future. Section 2.1 reviews the history of UAVs and shows the evolution of the vehicles from their genesis. This section also walks through some of the research currently occurring in the arena of swarm control algorithms. Section 2.2 explores the past and present sensor technologies used to collect information of various types. Section 2.3 reviews the state of research in sensor database technology.

2.1 UAV Swarming Technology

Swarming technology brings together several fields of research. First and foremost is the UAV. These vehicles and their on board sensors are the tip of the spear in intelligence gathering technology. Controlling the movement of UAVs are a host of complex algorithms, or computational models that interact with the agents and determine the paths flown by the individual aircraft.

2.1.1 UAV History. UAVs are not a new concept. From the early times of flight, engineers have been designing and building remotely piloted aircraft to perform a multitude of missions. In 1863 Charles Perley, an inventor from New York designed an unmanned aerial bomber constructed from a hot air balloon (Fig. 2.1) [8]. A timer was set to release a bomb after a certain mount of time in flight. As expected, this proved dangerous and unreliable. In 1888, Corporal William Eddy of Colorado took hundreds of surveillance photos from a kite during the Spanish-American war. This marked the first intelligence gathering missions of UAVs [8].

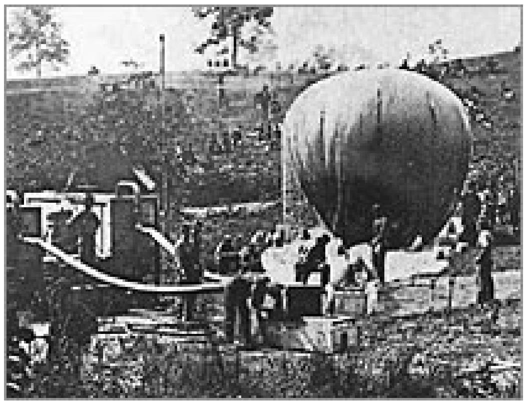


Figure 2.1: Perley's Aerial Bomber

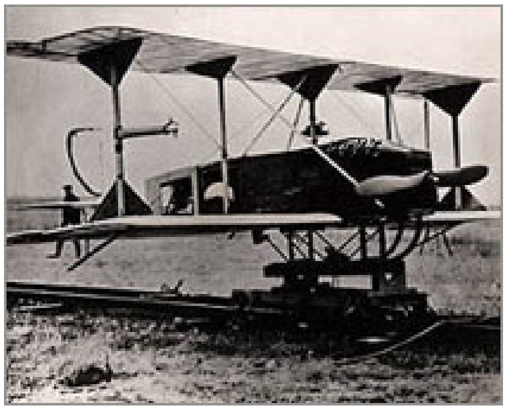


Figure 2.2: Sperry's Aerial Torpedo

During the first World War, the automatic gyroscope stabilizer was invented by Cooper and Sperry. This allowed an unpiloted aircraft to fly straight and level and was key in the developing the first radio controlled UAV, a converted U.S. Navy Curtiss N-9 trainer (Fig. 2.2). The aircraft was fitted with 300 pound bombs and had a range of 50 miles, but never saw combat [8].

In the 1930s, the British Royal Navy used the DH.82B Queen Bee UAV (Fig. 2.3) for aerial target practice. These radio controlled spruce and plywood planes could fly at 17,000 ft a distance of over 300 miles at 100 mph [8].

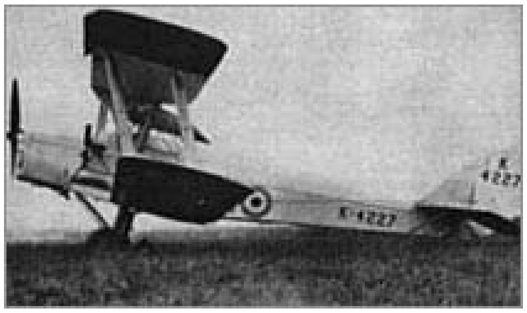


Figure 2.3: DH.82 Queen Bee UAV

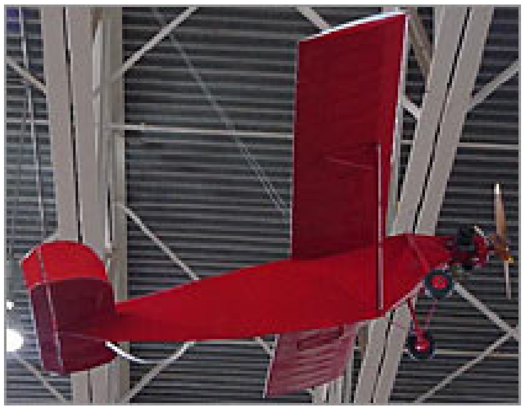


Figure 2.4: Radio Plane

In 1939, the beginnings of what is today known as Northop Grumman was formed by Englishman and actor Reginald Denny. He and a team of engineers from Lockheed worked on developing large remotely piloted airplanes for the U.S. military known as radio planes (Fig. 2.4). Over the years, the government bought thousands of the planes at about $600 each and used them to train anti-aircraft gunners [8].

During World War II, the Germans designed the Fieseler Fi-103, better known as the V-1 (Fig. 2.5). Using a thrust pulsejet engine, this vehicle could fly at speeds of 470mph for 150 or more miles while carrying a 2000 lb bomb [8].

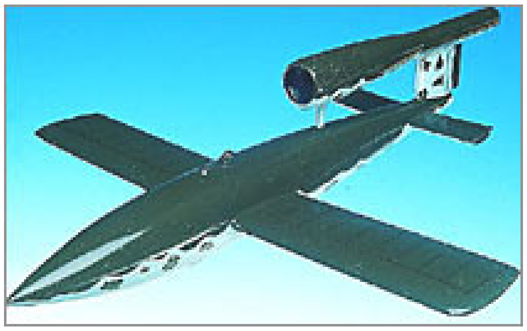


Figure 2.5: V-1 UAV

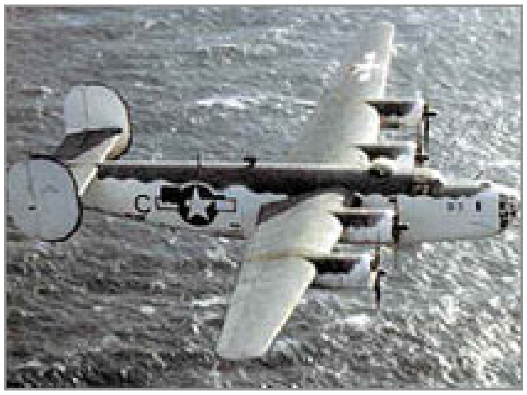


Figure 2.6: B-17 UAV

To counter the threat from the V-1, the U.S. Navy developed UAVs that could destroy V-1 launch sites. The modified PB4Y-1 Liberators and B-17 Bombers (Fig. 2.6) could carry 25,000 lbs of explosives directly to V-1 launch sites via remote control and television guidance systems [8].

During the early 1960s, the Air Force began to investigate stealth technology. The result was the AQM-34 Ryan Firebee (Fig. 2.7). The Firebee proved itself a capable reconnaissance aircraft, and between 1964 and 1975 more than 1,000 of the planes flew over 34,000 operational surveillance missions over Vietnam. Israel purchased twelve Firebees from the U.S. in 1970 and used them in the 1973 Yom Kippur War as both reconnaissance vehicles and decoys. In 1971, after an Air Force RC-121 communications intelligence (COMMINT) plane was shot down, a modified and redesignated version of the Firebee, called the SPA-147 (Fig. 2.8), was developed. This UAV was capable of carrying a 300 lb camera at 60,000 ft for eight hours, making it the first long-haul UAV to provide COMMINT at high altitude [8].

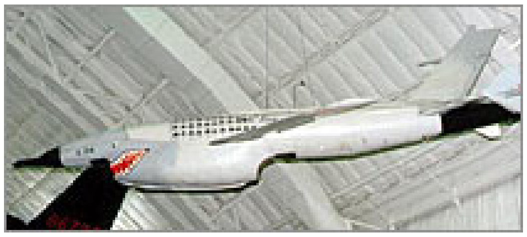


Figure 2.7: AQM-34 Ryan Firebee

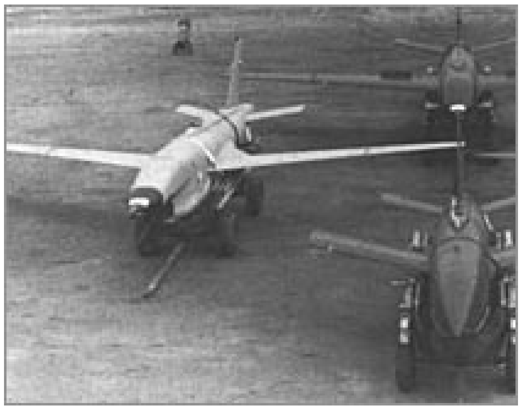


Figure 2.8: SPA-147 UAV

In the 1980s, the Israelis developed the Scout (Fig. 2.9) and Pioneer (Fig. 2.10) UAVs. These light UAVs proved valuable during the Bekka Valley conflict between Israel, Lebanon and Syria. The U.S. military was duly impressed and purchased 20 of the Pioneers. Today these work hand in hand with JSTARS aircraft to confirm high priority mobile targets seen on SAR imagery [8].

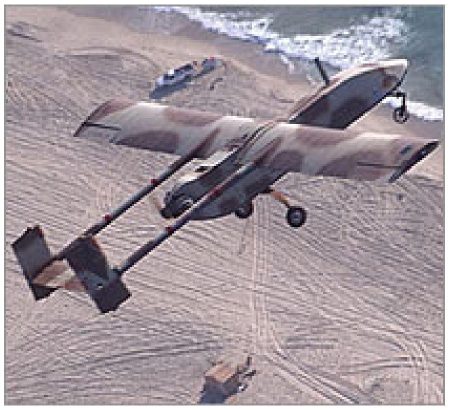


Figure 2.9: Israeli Scout UAV

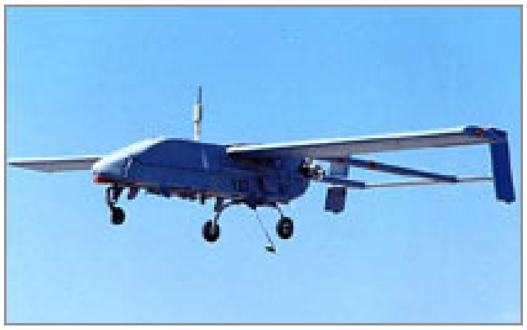


Figure 2.10: Israeli Pioneer UAV

The UAV that has made a name for itself in the past 7 years is the RQ-1 Predator (Fig. 2.11). Now virtually a household name, the Predator has been used in the Balkans, Afghanistan and the Middle East. With a 450-mile range and the capability of loitering on station for fourteen to sixteen hours, it can provide imagery via high definition color television, SAR and IR cameras. Initially the Predator was designed strictly as a surveillance plane, but has recently been outfitted with Hellfire anti-tank missiles which have successfully engaged enemy targets [8].

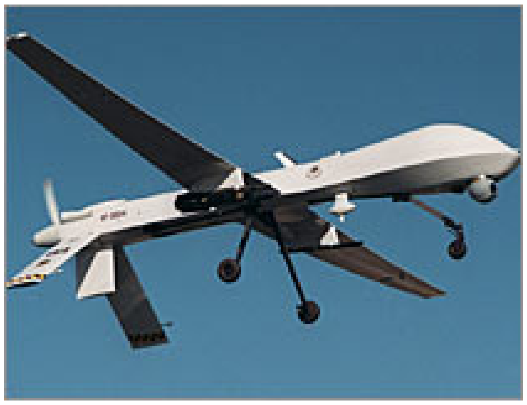


Figure 2.11: RQ-1 Predator UAV

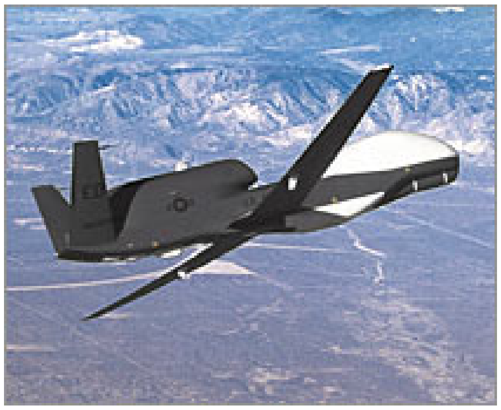
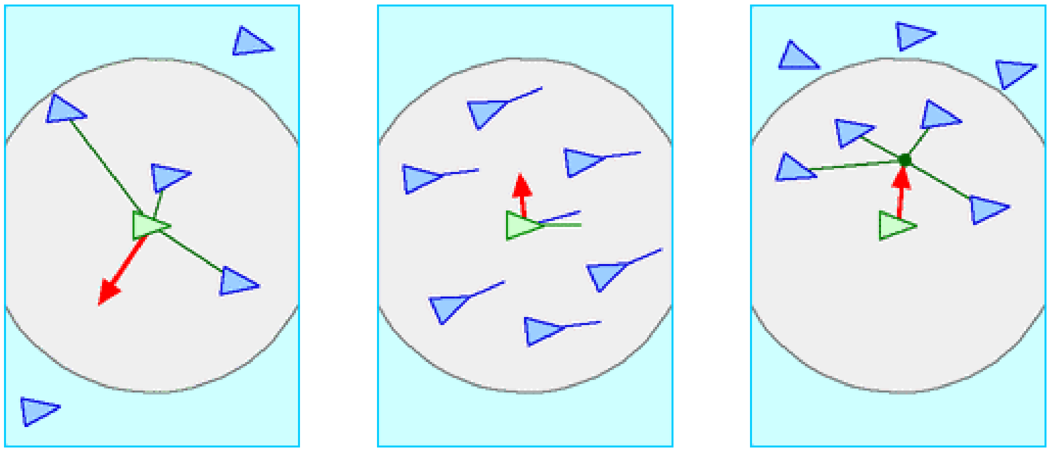


Figure 2.12: RQ-4 Global Hawk

The newest generation in high performance UAVs is the Northrop Grumman RQ-4 Global Hawk. This UAV, with its 116 ft Wingspan and efficient jet engine can fly at 65,000 feet for over 36 hours with a range of 13,500 nm . Its integrated sensor suite consists of SAR, IR and electro-optical (EO) sensors with either the EO or IR sensors able to operate simultaneously with the SAR sensor. In addition the SAR has a ground moving target indicator capable of calculating a target's position and velocity and then sending that information to ground stations. [1].

Current research is beginning to focus on mini and micro UAVs. The ability to fly tens, hundreds or even thousands of UAVs in a swarm requires that each individual vehicle be compact, easily shipped, set-up and launched, have sophisticated sensor technology on board, be stealthy, and able to loiter over areas of interest for long durations. These requirements individually are relatively easy to handle, but taken together represent a significant challenge for the designers and acquisition offices.

2.1.2 Control Algorithms. As mentioned, the idea of a swarm of UAVs is to send a group of unmanned planes (*group* could be as few as two or as many as several thousand) out on reconnaissance missions. This swarm of aircraft is guided by some routing mechanism that steers the swarm through a generally pre-planned mission profile. The individual members of the swarm themselves are autonomous vehicles that move within the swarm according to a set of rules that define movement based on rules defined by Reynolds [11]. As a single agent moves, it is constantly aware of the proximity of the surrounding agents. According to Reynolds, three basic control rules govern the movements of the agents within the swarm. *Separation* is used to avoid a collision with other nearby agents (Fig 13(a)). *Alignment* attempts to match velocity and steer to a heading consistent with nearby agents (Fig 13(b)). *Cohesion* works to keep the agents in close proximity to one another by steering the agents toward the average position of the local agents (Fig 13(c)).



(a) Separation (b) Alignment (c) Cohesion

Figure 2.13: Basic swarm control factors, Reynolds [11]

2.2 Sensor Technology

A critical piece of the intelligence gathering equation is the use of sensors for such ventures. Today's UAVs are generally fitted with various types of sensors designed to scan the terrain for targets of interest. The basic types of sensors used are video, infrared (IR), chemical, acoustic and synthetic aperture radar (SAR). These five types of sensors are discussed in the following sections.

2.2.1 Video. Video sensors are by far the easiest of the sensors to understand. Using small high-definition television cameras mounted on moving platforms called gimbals, these cameras scan the ground in either a pre-determined pattern, or search for particular targets from a list of locations. As the vehicle moves within range of the targets, the computer directs the field of view of the camera toward the target. It is also possible for the camera to be directed from an operator on the ground. The images captured by these cameras are understandable as viewed by the naked eye and reside in the visible light spectrum ($700 - 400\ nm$) the same as a television camera, meaning that cloud cover, foliage and darkness will obscure the target.

2.2.2 Infrared. Infrared sensors operate in the manner same as video cameras, except they detect a different part of the electromagnetic spectrum. Infrared radiation lies just below the visible spectrum, from $.01\ cm - 700\ nm$. These sensors detect thermal differences across their field of view and create an image from these differences. The advantage of these sensors is the ability to "see" at night. Darkness has no effect on the sensor since the part of the spectrum received at is not significantly affected by the change in ambient light.

2.2.3 Chemical. Chemical detection sensors are an important part of the information gathering grid. These sensors chemo-mechanically "sniff" the air for anomalies in air quality. The sensors can detect trace amounts of chemicals in the air and determine the lethality of the ambient airspace. Conventional chemical sensors systems are built around substance-specific sensors that filter out interfering stimuli and indicate whether the offending chemical is present [12]. As the world looks upon terrorists that have a vast array of lethal technology at their fingertips, the prospect of unleashing deadly chemical weapons becomes more of a reality. By employing these sensors on board swarms of UAVs, the possibility exists for detecting the release of chemicals or the production of chemicals in an area of interest.

2.2.4 Acoustic. Acoustic sensors are used to passively track airborne and ground vehicles without line of sight restrictions. A wealth of information can be extracted from these sensors, including range, bearing, velocity and classification. While most of these measurements have been made using ground based sensors, the employment of these sensors on board UAVs is possible. Using noise canceling technology, the sound produced by the UAV itself is subtracted from the noise field and the data collected from the sensor about happenings below are evaluated. By combining the information from the sensors with the movement of the UAV it is possible to track ground targets, or locate them based on their sound output. When used in multiple UAVs the sensors act as an array. Over a battle field, both continuous signals (moving vehicles) and impulse signals (gun shots) can be monitored, classified and located.

2.2.5 Synthetic Aperture Radar. Synthetic aperture radar is now a common sensor on board many military intelligence gathering aircraft. The basic principle of SAR is the emission of energy from a moving platform across the ground. As the vehicle moves, the radar waves are transmitted to the ground and subsequently reflected. The antenna on board the aircraft receives these reflections and pieces the series of reflections together to form an image. The width of the image strip depends on the capabilities of the SAR system, but generally will decrease in width with an increase in vehicle speed. This is primarily due to limitations in processing. Depending on the SAR system used, various resolutions can be achieved. Typically resolutions are between 0.3 m and 1.0 m, meaning that the smallest discernable object in the picture is of that size or larger.

2.3 Sensor Database Technology

Sensor database technology is a growing field that is still defining standards of concepts and practices. The following sections explore the differences between traditional data and sensor data as well as some of the different methods needed for accessing this type of information.

2.3.1 Sensor Data Storage.

The storage of sensor data requires a large volume of information storage capability as well as a robust communication system to move the information from the sensor to the storage facility. In the case of swarming UAVs, there are several (possibly hundreds or thousands) sensors that are constantly gathering data that needs to be stored. The database must contain both stored data and sensor data [6]. The difference between the two is critical to understand. *Stored data* is the set of sensors that contribute data to the database together with the characteristics of the particular sensors. This provides a base for understanding the type of data that the particular sensors will be contributing to the *sensor data* store. The *sensor data* is the set of data actually collected, analyzed and sent to the database by the sensors. These are the measured values that are of interest to the analysts. Both sets of data are needed by the end user. Accurate analysis of the sensor readings can only be obtained by understanding the capabilities of the sensors. Anomalous readings on a sensor can be attributed to several factors including power surges, radio frequency (RF) interference, rapidly changing phenomena and sensor measurement range limitations. By understanding the capabilities of the sensors, or having the ability to retrieve the capabilities from the same data store as the sensor data, the user can draw conclusions about whether the readings seen are possible from the sensor.

2.3.2 Sensor Data.

Sensor data is fundamentally different from traditional flat file data. Sensor data are generated by signal processing functions [6]. Because data is arriving constantly, there are a number of parameters that need to be monitored such as those discussed in section 2.4.

Time plays a critical role in differentiating “pieces” of sensor data. Consider a stream of IR data from a UAV camera. Without knowing the time and place the data was collected, it could be difficult to determine what the image was showing. Several parameters that define a data stream can be defined by time. The methods proposed in this research are time dependent and rely heavily on the availability of time data as part of the information to be retrieved from the data store.

In addition to time, the geographic location of the sensor becomes a factor when sensor data is gathered from moving entities. In an UAV swarm sensor database, the locations of the UAVs are dynamic and must be taken into consideration when looking at the data gathered. As an example, suppose the time sequence of infrared data on a particular target shows a dramatic decline in measured temperature over a short period of time. Several explanations for this are possible, one being that the sensor moved from the side of the target lit by the sun to the other side that could have been sitting in the shade for hours. Without knowing the location of the sensor relative to the target in time (as well as the position of the sun, which can be determined by the time of day), it would be difficult to extract this particular explanation. To complicate the picture, there are only discrete time references to the positions of the sensors. Perhaps updates to the position of the UAV occur every five seconds, while the sensor is collecting data at a rate of 10Hz. This means that there are 49 measurements for which there is no precise positional data. This may seem like a small problem, but consider that with a UAV traveling at 120Kts (210+ feet per second) the average movement of the UAV is 21 feet between measurements. At a high altitude, this may not be too much of a problem, as the field of view of the sensor sees a large area. At low altitude, where the sensor may be looking at an area only a few times larger than the distance the vehicle travels in a second, this can be a problem.

Finally, sensor data is unproven data. That is, the data has yet to be corroborated against any observed norms, and as such, may not necessarily be implicitly trusted without cross validation. Corroboration of the data by several sensors is preferred in order to increase the confidence that sensed values, as reported from the sensor, are correct. A history of known good values from a particular sensor is also confidence inspiring.

2.3.3 Query types. The types of queries asked of a sensor database tend to be about a wide spectrum of spatial-temporal information needs. Consider the need to know data from specific UAVs that were near a certain target at a certain time. This combination query must follow a specific order for the correct data to be extracted. First, the set of UAVs that passed *within a certain range* of a target must be taken from the database. Second, that set must be scoured for the agents that fulfill the *time* requirement. Third, the sensor data needs to be taken from the final resulting records and analyzed. Pfoser et.al. [10] call these types of combination queries *spatiotemporal* queries. The location (near the target) is the spatial part, while the time (during this part of the data run) makes up the temporal part. Figure 2.14, taken from [10], shows the concept. In this figure, the dotted cube represents the spatiotemporal range used to select the UAVs, and the line represents the trajectory of the center of mass of the swarm.

2.4 Mission Semantics

In a swarm of UAVs, the individual cameras may need to operate autonomously, recording not only the image seen through the lens, but a host of other parameters as outlined in Table 2.1, partially taken from [5]. As the information from the sensors is analyzed, algorithms must extract not only the data from the cameras, but the data about the position and flight dynamics of the UAV platform as well. Sensors are mounted on the UAV and their pointing data is relative to the platform. Without knowing the position and attitude data from the UAV, there is no way of knowing

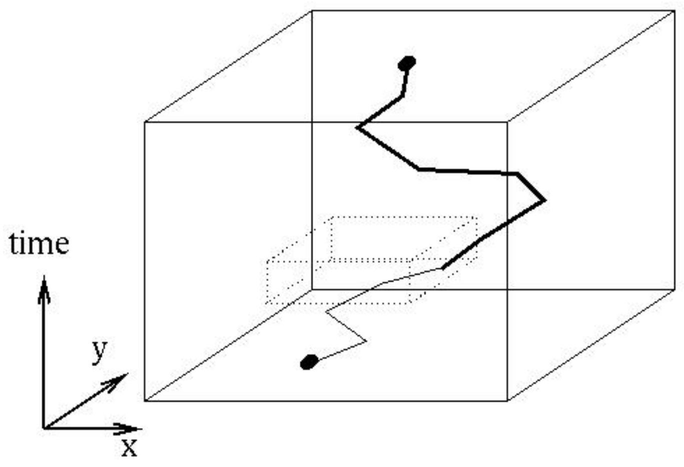


Figure 2.14: Graphical representation of a spatio-temporal query

exactly where the sensor is pointing at any given point in time. Separating the two types of data allows the analyst to decipher exactly where the camera is pointing.

2.5 Summary

Combining the sciences of UAV swarms and sensor technology offers the potential for unprecedented information gathering capability. The development of both of these fields has reached a point where this combination is on the near horizon. The ability to take full advantage of the capabilities of this type of system of lies in the development of algorithms and tools that automate the task of finding the specific data within the vast amounts of created data that allows decision makers to quickly make decisions that have the greatest impact on operational capability.

Table 2.1: Telemetry Data Fields and Definitions

#	Telemetry Element	Description
1	Current universal time	Universal Time Coordinated (UTC) hour of the mission
2	Camera center look point Latitude	Represents the geographic latitude of the center of the frame being recorded
3	Camera center look point Longitude	Represents the geographic longitude of the center of the frame being recorded
4	Camera Altitude	Represents the altitude above mean sea level (MSL) of the location being recorded
5	Platform Altitude	Current geographic latitude of the UAV
3	Platform Longitude	Current geographic longitude of the UAV
7	Platform Altitude	Current altitude above MSL of the UAV
8	Platform Heading	Direction of motion of the UAV from north based on 0-360 degrees clockwise
9	Platform Roll	Degree of roll the platform is experiencing
10	Platform Pitch	Degree of pitch the UAV is experiencing
11	Bearing	Compass heading of the UAV
12	Camera Depression	The angle from horizontal that the camera was pointing. +90 is straight up, -90 is straight down
13	Zoom Factor	The percentage of available zoom used at the time

3. Methodology

As more emphasis is being placed on swarming UAVs for intelligence gathering, tools need to be developed that allow for retrieval of data efficiently. The natural inclination of the planners is to believe that if there are more "eyes" on the target, there should be more data available to make decisions with. Certainly with the possibility of corroborating data (that is, the ability to cross-check readings from several sensors looking at the same target at the same time), there exists a potential for making decisions using highly relevant and reliable information. Extracting the times and places where relevant data should be found is a matter of analyzing the swarm movement in time over the target field. By looking at the swarm from a geometric standpoint, an *information quality expectation* can be created that points the analyst to the data most likely to be in space, time, and quality consistent with the need.

Berridge [5] looked at this issue from the stand point of a single UAV mission. His methods pinpoint relevant data from the telemetry sent from the sensor on board the aircraft. His methods allowed the user to enter query parameters from a graphical user interface (GUI) and pick out the segment of a UAV video stream containing the relevant data based on such query parameters as time of day, camera angle, sun angle, distance to target, and platform altitude. When the UAV is but one unit of a swarm of possibly hundreds, and when each UAV could possibly carry multiple sensors, methods are needed that first pare down the possible UAV feeds to a manageable number from which to query the telemetry.

3.1 Overview

This work centers on the development of algorithms and techniques used to make decisions about data relevance based on target location and the movements of the swarm in time over the target field, as well as an analysis program that demonstrates these capabilities. Section 3.2 defines *information quality expectation* and the methods developed to support the discipline. Distance and bearing algorithms, centers of mass, target-centric information expectation and target profiling are introduced and discussed. Section 3.3 introduces the concept of the convex hull and explains how this geometric construct is integral to the modeling of information expectation. By understanding the convex hull, calculations of swarm density, sensor density, and coverage area are calculated and contribute to the overall quality expectation.

3.2 Information Quality Expectation

Webster [13] defines *information science* as "the scientific study of the gathering, manipulating, classification, storage and retrieval of recorded knowledge." In the practice of aerial surveillance as an element of ISR, analysts are constantly gathering information about the actions of the ground based targets of interest. This information comes from myriad sources and in numerous forms. With the advent of swarming UAVs on the horizon, the information volume available to analysts promises to grow significantly. The concept of *information quality expectation* can be built on the definition of *information science* to include the "expectation that, based on an analysis of swarm position in time, information highly relevant to a need can be efficiently extracted from the recorded knowledge".

This report focuses on modeling the *expected information quality* from a swarm of multi-sensor laden UAVs. In particular, the swarm is considered from a geometric standpoint to determine such parameters as sensor coverage, swarm density, sensor coverage density and center of mass of the swarm. Center of mass, in particular, may apply to the swarm as a whole as well as subsets of the swarm based on sensor capabilities. By coupling computed parameters with data showing the proximity of the swarm to various targets, it is possible to focus on a subset of agent nodes with certain sensors from which to request data pertaining to the target of interest.

The information about the identified agents will show that a critical mass of data is available about a target, and that that mass of data can be corroborated to ascertain whether readings from the sensors are true. Relevance, as mentioned in section 1.3 is the key to using the methods developed here. By function, UAV swarms collect large amounts of information. An expectation for specific information, relevant to decision quality needs of the analyst, requires a methodical analysis of the swarm's position in space-time while gathering data. To facilitate decision-quality data extraction from a swarm's data store; we propose algorithms which:

1. Determine times where agents had "eyes on target".
2. Calculate the time-density of coverage. As time progresses, there are times when the coverage density, or number of sensors per unit area are higher. At these times, the probability of detection is higher. If these times coincide with the swarm's close proximity to a target, the expectation becomes higher that useful information has been obtained.
3. Establish the possibility of corroboration of sensor readings. By comparing the time synchronized data, certain times emerge where multiple similar sensors were looking at a target concurrently. These times allow for corroboration of the data taken by the sensors. Anomalies between sensors can highlight problems with one or more sensors, or show differences in the accuracy with which a target is detected from different angles via different sensors
4. Calculate the duration of overage. Depending on the route particular UAVs take, some will spend more time in the vicinity of a target than others. These represent opportunities for increased data, and consequently the chance to learn more about the target.
5. Determine UAV position relevant to the target. UAV position with respect to the target will determine limits on various sensor capabilities. Sensor readings will change based on distance and bearing to the target. Knowing the relative

positions of the UAV and target, analysts can make more accurate conclusions about the meaning of sensor data.

6. Compute cross-sensor data validation. UAVs carrying multiple sensors have the capability of using readings from one sensor to validate readings from a second sensor (e.g. an infrared sensor may be able to validate an image taken by a video sensor in dim light). In the absence of other UAVs with which to corroborate data, this technique would allow at least some way to check sensor reading validity.

3.2.1 Distance and Bearing Algorithms. Planar distance and compass bearing to the target are critical information that need are gathered and analyzed when attempting to determine information quality expectation. The combination of sensor field of view and UAV distance-to-target drive this expectation. By knowing the planar distance a UAV is from a target, as well as the field of view of the sensor, it is easy to determine the likelihood that the sensors on board the UAV collected any useful data.

We can apply Pythagoras' Theorem to calculate the distance (D) between a given target and UAV given their latitude and longitude as shown in Equation 3.1 and Figure 3.1.

$$D = \sqrt{(LAT_{uav} - LAT_{tgt})^2 + (LON_{uav} - LON_{tgt})^2} \tag{3.1}$$

Where:
LAT_{uav} = latitude of the UAV
LON_{uav} = longitude of the UAV
LAT_{tgt} = latitude of the target
LON_{tgt} = longitude of the target

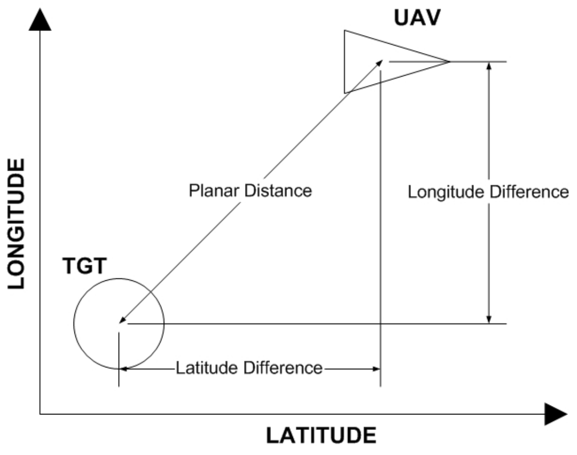


Figure 3.1: Planar distance between target and UAV

As stated, compass bearing from the target to the UAV is also an important metric. Sensor readings can change according to the view of the target by the sensor. Consider a chemical cloud emanating from a target as in Figure 3.2. With the prevailing winds coming from the north, chemical sensor-laden UAVs flying north of a line drawn from west (bearing 270) to east (bearing 090) has little chance of sensing the chemicals, while UAVs flying south of that line have a better chance of detecting the cloud the further south they move. The UAVs not detecting any chemicals are indicated in black, the ones that are about to sense something or may have already sensed trace amounts are gray and the ones clearly in a position ot detect chemicals are indicated in white. Calculation of the bearing (Θ) is done using Equation 3.2.

$$\Theta = \arctan(\frac{(LAT_{uav} - LAT_{tgt})}{(LON_{uav} - LON_{tgt})}) - 180 \tag{3.2}$$

Combining the distance and bearing calculations allow the analyst to extract conclusions about the types of readings expected from sensors viewing the target.

3.2.2 Swarm Centers Of Mass. The centers of mass of the swarm and the sub-swarms play an important role in the research. By modeling the geometry of the

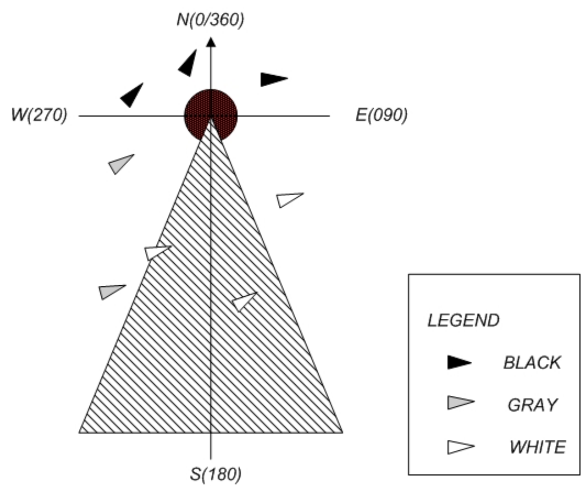


Figure 3.2: Expected data collection opportunity value as a function of sensor's bearing to target

center of mass of the swarm, it becomes possible to predict times and locations where the high-quality data collection could have taken place. While knowledge of center of mass alone cannot ensure accurate predictions about data collection capability, the combination of center of mass, coverage area, and sensor density at correlating times presents an opportunity for more accurate predictions of times when relevant data may be available.

Calculating the centers of mass of the main swarm and the sensor family sub-swarms is relatively straightforward. The center of mass of a swarm is a point (LAT_{CoM}, LON_{CoM}) determined by averaging the latitude of the points and the longitude of the points using the following formulae:

$$LAT_{CoM} = \frac{\sum_{i=1}^{n} LAT_i}{n} \qquad LON_{CoM} = \frac{\sum_{i=1}^{n} LON_i}{n} \tag{3.3}$$

For this work the center of mass of the main swarm as well as the sub-swarms needs to be calculated at each time step to allow tracking of the centers of mass throughout duration of the swarm flight.

3.2.3 Target Centric Information Expectation. A target on the ground, whether stationary or moving, represents an opportunity for data collection. Few targets are invisible to today's high tech sensor suites. By focusing the search for relevant data in a methodical and logical way, the chances of retrieving the data needed are greatly increased. The quick retrieval of relevant data about a target allows the decision makers to compress their decision time frame and effect timely results. Target *profiling* gives the analysts a view of the coverage the swarm geometry affords a particular target.

3.2.4 Target Profiling. Target profiling is the process of analyzing the movement of the entire swarm in time and determining when to expect the greatest amount of relevant data about the target. Essentially this answers the question "When can I expect to have the best sensor information on this target based on the movement of the swarm?." As a UAV swarm moves, the proximity to a stationary target of each node within the swarm changes over time. As the individual UAVs move closer to the target, their opportunity to collect data on the target increases. The concept of profiling a target is to look at the swarm at each distinct point in time and determine, based on the positions of the swarm members, the likelihood of relevant data collection on that target.

3.2.4.1 Profiling Definition. The profile of a target is a plot of sensor-distinct quality values (Sec 3.2.4.2) over time. The profile shows data collection likelihood for each of the four sensor types (video, infrared, chemical, and acoustic) for the duration of the swarm flight. At each distinct time slice, a snapshot of the swarm is taken. For each family of sensors, the aggregate quality value for the time slice is the sum of the collection-quality scores assigned to the UAVs that have sensors of that family.

3.2.4.2 Quality Values and Scores. Each UAV is assigned a quality score based on its distance from the target. As the distance from the target increases, the quality score decreases. The area of interest is divided into equal sized rings concentric with the target. The number and size of the rings are based on what we term a *fog* index (Sec 3.2.4.3) specified by the user. As the number of rings (r) increases outward from the target, the quality score of the ring decreases by $\frac{1}{2^{r-1}}$. Figure 3.3 shows an arbitrary area of interest with a single target. The area of interest is divided into the rings concentric with the target. As the rings increase in number outward from the target, their value decreases according to the above formula. Using this formula, the center ring ($r = 1$) will always have a value of 1 based on $\frac{1}{2^0}$

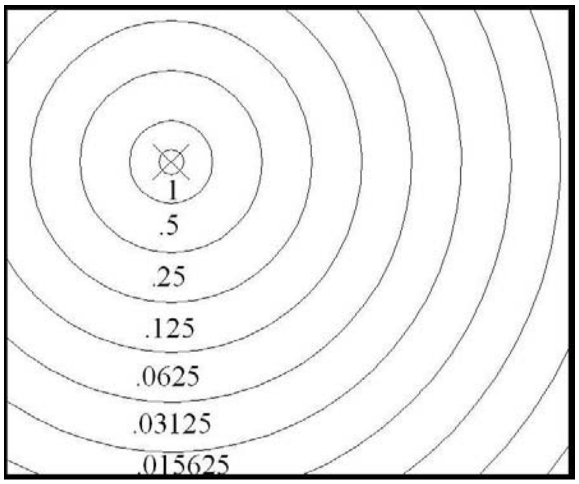


Figure 3.3: Distance based quality values

3.2.4.3 Fog. For this work, the term "fog tolerance" refers to a value set by the user to determine the precision of a target profile. The fog tolerance is an integer between 1 and 50. As the fog tolerance (D_{fog}) increases, the number of concentric rings (r)around the target decreases as in Equation 3.4. As the tolerance for fog increases, so too does the tolerance for a less precise profile. With fewer rings, the profile shows a more general view of the likelihood of information. More rings only return higher values when the UAVs come relatively close to the targets where there is a higher liklihood of successful data collection. The number of rings ranges from 10 to 500.

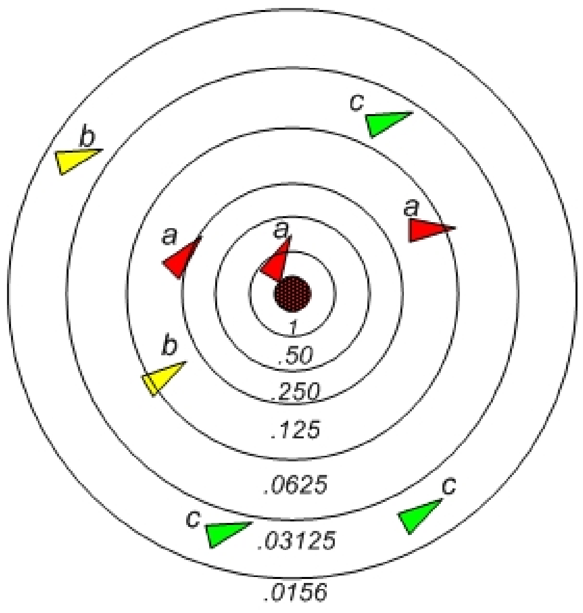
$$r = 5 * \frac{100}{D_{fog}} \tag{3.4}$$

The distance (d) between the rings is based on the average of the "length" and "width" of the area of interest (see sec. 4.3.1.2 pg. 44 for area of interest discussion) (Equation 3.5). The average of the latitudinal distance and longitudinal distance is determined and divided by the number of rings. The resulting number is the distance between consecutive rings. The distance to the outer ring is always set as the maximum distance from the target to the outer boundary of the area of interest. In this sense, ring "density" reveals a gradient of quality score values decreasing away from the target.

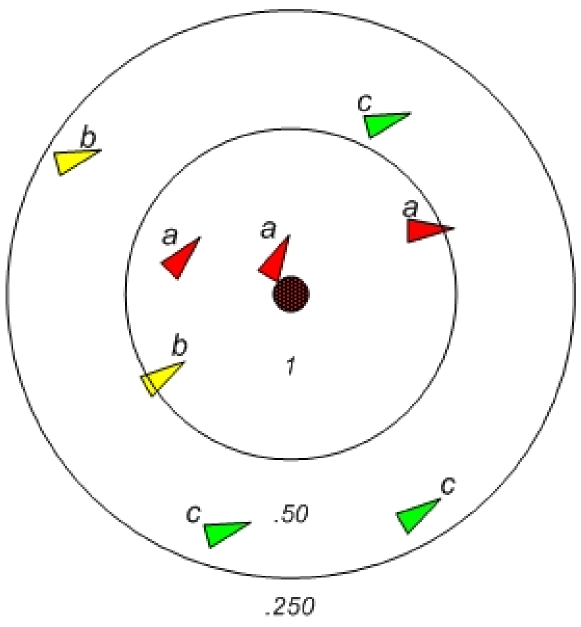
$$d = \frac{\frac{(LAT_{max} - LAT_{min}) + (LON_{max} - LON_{min})}{2}}{n} \tag{3.5}$$

Figure 3.4 illustrates the concept of profiling. Three types of UAVs (a, b and c) are shown. They are in the same position relative to the target on both diagrams, however the fields are divided into different numbers of regions. Figure 3.4(a) has a lower fog tolerance value and thus has more discrimination between the distances from the target than does Figure 3.4(b). The resulting profile from 3.4(a) has a more realistic quality values, as the higher values are assigned only when the UAV passes closer to the target.

A couple of assumptions are made when constructing this model. First, altitude is not taken into consideration. UAV swarms are currently modeled as planar, so there are no differences in the altitudes of the swarm members. Second, this is intended to model the *likelihood* of sensor information, and not the sensor information itself. Because the range of capabilities of sensors varies widely, proximity to the



(a) Low fog tolerance



(b) High fog tolerance

Figure 3.4: Visual representation profiling concept

target is the only consideration. Modeling individual sensor capabilities, and the manner in which data collection takes place are discussed in the future work chapter.

Table 3.1 shows the resulting quality scores for this scenario. The values are calculated by summing the values of the rings over which the UAVs are at the point in time.

Table 3.1: Profile quality scores from Figure 3.4

Fog Tolerance	"a" UAV Value	"b" UAV Value	"c" UAV Value
Low - 3.4(a)	$1 + (2 * .125) = 1.250$	$.125 + .031 = 0.156$	$.0625 + (2 * .03125) = .125$
High - 3.4(b)	$3 * 1 = 3.00$	$1 + .50 = 1.50$	$3 * .50 = 1.50$

The motivation behind allowing selectable fog tolerance values is the ability to simulate different information quality expectation scenarios. If updates to the position of the UAV are made at increasingly longer time intervals, determining the exact distance from the UAV to the target at smaller time intervals can be difficult. One can interpolate the distances and attempt to get an accurate profile from those intermediate values, or assume the UAV is stationary between position updates. By using the variable fog profile, the differences in information quality expectation can be seen and modeled.

A practical application of this “dial-a-fog” capability is the simulation of Global Positioning System (GPS) inaccuracies and variances in GPS credibility. GPS positional accuracy is based on the number of satellites seen by the UAV. If the number of satellites seen is less than needed for accurate positional data, the telemetry will report inaccurate UAV position information. The variable fog profiler can be used to model this and reflect the differences in information quality expected from the swarm.

The concept of information quality expectation is more than just calculations of distances, bearings and profile values. Geometric analysis of the swarm plays an important part in determining: 1) when opportunities exist for data corroboration and 2) probabilities of data collection. The following section outlines methods for analyzing the swarm from a geometric standpoint.

3.3 Convex Hulls as a Swarm Boundary Marker

When projected on a plane, the elements of a swarm can be considered vertices in a graph. These vertices move over the area of interest, changing the shape of the area covered as the swarm moves according to its control algorithms. We are interested in the shape of the vertices as a means to determine the density of the swarm coverage. We use the convex hull for this purpose. The convex hull of a set of points is the convex polygon created by connecting a subset of the points in a graph such that the hull formed by the connecting line segments is, at every turn, in the same direction. This has the effect of drawing a set of line segments around the set of points and touching only the outermost points in the set. For this research, the Quickhull algorithm [4] is used.

Figure 3.5 helps illustrate the algorithm. First a line is drawn between two points (A and B) known to be on the convex hull by their extreme positions on the X-axis. Each point on one side of the line is then checked to see if it forms the largest triangle when combined with A and B. When one is found (C) the points inside the triangle are eliminated (E, F, G, and H). The procedure is repeated on the other side of the original line where (D) is found and (I, J and K) are eliminated. This process is recursively repeated using the edges of the drawn triangles. The result is the convex hull.

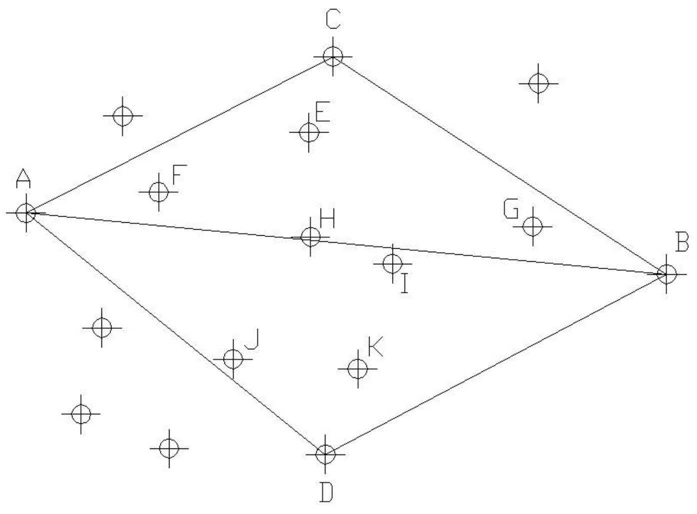
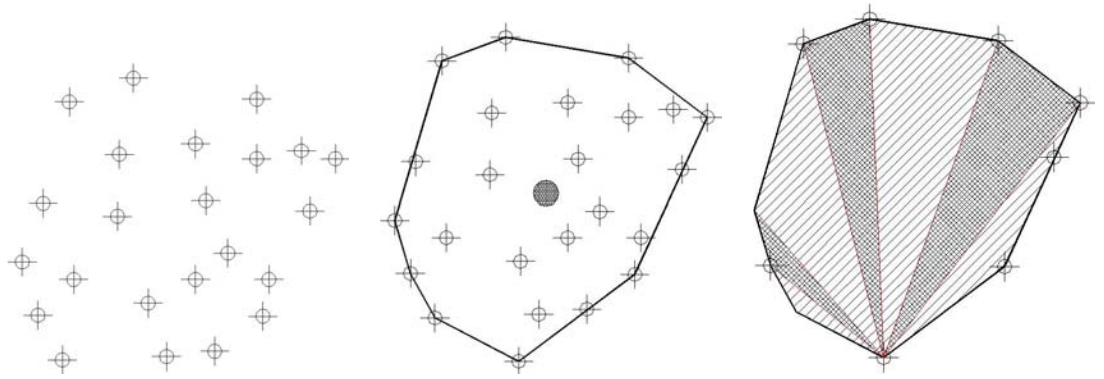


Figure 3.5: Depiction of the Quick hull algorithm

As a practical example, suppose some number of nails are pounded into a piece of wood. When a rubber band is stretched around the perimeter of the set of nails, the shape the rubber band takes is a convex hull. Figure 3.6 illustrates this concept in two dimensions. Three-dimensional convex hulls are applicable for 3-D swarming objects, but for this work, convex hulls are limited to two dimensions. Exploration of the geometric properties of the convex hull is an integral part of this research.

Convex hulls are used to compute several information expectation forming parameters from a swarm. The area inside the convex hull is made up of adjacent triangles, making the calculation of the area trivial. Having this information allows



(a) Points in a plane (b) Convex Hull and center of mass (c) Area of convex hull

Figure 3.6: Illustration of the convex hull, center of mass and area

for the computation of numerous swarm characteristics that define the expected information returnable from the swarm.

3.3.1 Sub-Swarms. Prior to discussing some important geometric properties of a swarm, it is useful to define sensor-based sub swarms. In this work, the UAVs have various sensors assigned to them. The main swarm can be subdivided into sub swarms based on the sensors on board the UAVs. For illustrative purposes in the simulation used in this work, sensors are assigned to the UAVs based on a user determined percentage. If, for example, 50% of the swarm nodes are to have IR sensors, and there are 100 UAVs, 50 randomly selected UAVs will be assigned IR sensors. This holds for all sensor types. The outcome of this is that some UAVs are going to have multiple sensors on board, while others are assigned none unless at least one of the sensor assignment values is 100%. This means that membership in several sub swarms by a particular UAV becomes more likely as the percentage of the swarm assigned to various sensor types increases.

3.3.2 Swarm and Sensor Densities. *Swarm density* (Δ_{Swarm}) refers to the number of UAVs within the area of the swarm convex hull (Equation 3.6) .

$$\Delta_{Swarm} = \frac{n_{UAVs}}{Area_{SwarmConvexHull}} \tag{3.6}$$

Knowing this density and the field of regard of each sensor allows analysts to determine the total coverage for the swarm. By monitoring the area of the convex hull of the whole swarm, the swarm density can be determined at any reporting instant. Likelihood of corroboration can also be linked to swarm density. The tighter the swarm is, the more likely there are multiple UAVs with "eyes on target" at the same time. By looking at swarm density, it is possible to determine the probability of recording relevant corroborable data (Sec. 3.3.3).

Sensor density (Δ_{Sensor}) is defined as the number of UAVs with a given type of sensor divided by the area of the convex hull of that sub swarm (Equation 3.7).

$$\Delta_{Sensor} = \frac{n_{SensorFamilyUAVs}}{Area_{SensorFamilyConvexHull}} \tag{3.7}$$

The area is determined by the the convex hull of the points defined by the locations of the UAVs with the sensors of interest. Figure 3.7 illustrates the concept. In this figure there are 11 UAVs with similar sensors on board. The line segments represent the convex hull of the swarm. The area of the swarm is 177.6 square units (arbitrary), so the density is

$$D_{Sensor} = \frac{11}{177.6} = .062 \text{ sensors per (unit of area)}^2$$

As the perimeter of the points defined by the convex hull increases and decreases due to the dynamic nature of the swarm, there is a corresponding decrease and increase in sensor density. For example, if the area decreased by half, the sensor density would double.

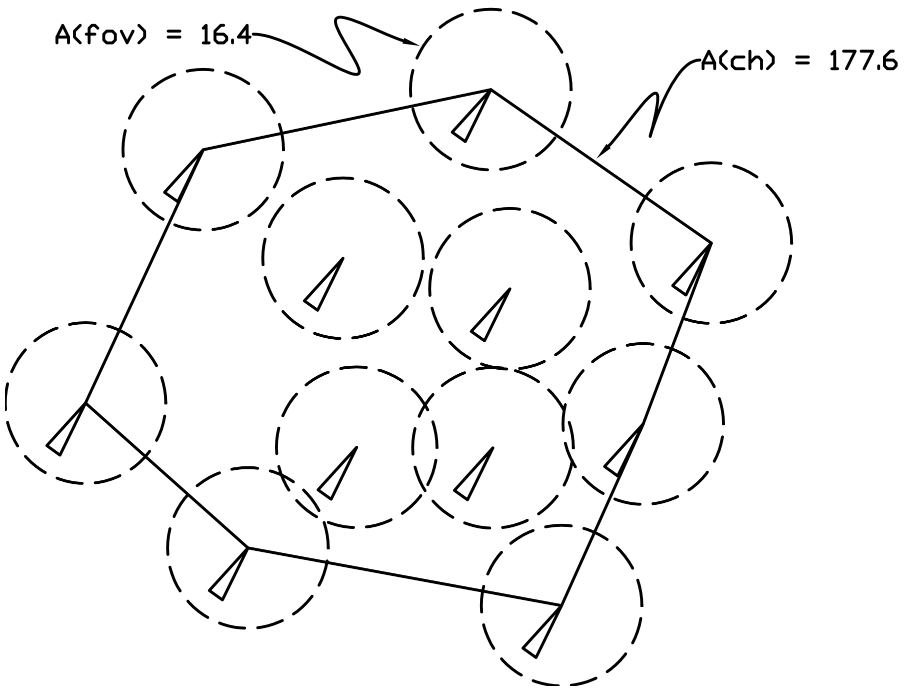


Figure 3.7: Illustration of swarm showing convex hull and sensor areas

3.3.3 Coverage. If the field of view of the sensors is known, an approximation of the probability of detection (P_{det}) of any given target (Equation 3.8) within the swarm boundary can be calculated. We estimate the probability by:

1. Finding the product of the number of sensors (n_{sens}) times the average field of view (FOV_{avg})of a sensor. This establishes the total area covered by the sensors.
2. Subtracting half of the product of the number of convex hull points ($n_{CHPoints}$) times the average field of view of each sensor on the hull. This is done to take into account the fact that for the points on the convex hull, half of the field of view of the sensor is actually inside the convex hull (Fig. 3.7).
3. Dividing the difference by the area of the convex hull (A_{ch}) .

Mathematically it is expressed as follows:

$$P_{det} \approx \frac{(FOV_{avg} * n_{sens}) - (\frac{n_{CHpoints} * FOV_{avg}}{2})}{A_{ch}} \tag{3.8}$$

For the configuration in Figure 3.7:

$$P_{det} \approx \frac{(16.4 * 11) - (\frac{7 * 16.4}{2})}{177.6} = .693(69.3\%)$$

If the swarm spreads out (increases the area inside the convex hull) the sensor density decreases as does the probability of detection. This calculation does not take into consideration the possibility for sensor coverage overlap. This number could be higher than one if the sum of the areas of the sensor fields of view is greater than the area of the convex hull. This happens when the swarm is flying in a very tight formation. Because this is a probability of detection only and not the number of times the target is detected, the actual value will never be higher than one. If the value calculates higher than one, it is capped at one.

3.4 Summary

Information expectation is not a stand alone metric that can be quantified. Instead, it is a concept that requires several pieces of quantified information combined to lead analysts to conclusions. Studying geometry of a swarm of multi-sensor-laden UAVs over time allows analysts to make decisions about when the best information was obtained about the target field of interest. Target profiling gives the analyst a sense of when quality information is available. Target coverage queries identify the UAVs that produce the high values in the target profile. Distance and bearing calculations tell where the sensor was in relation to the target, possibly explaining anomalous sensor readings. Swarm and sensor densities establish the possibility of corroborating data between sensors. By understanding the dynamics of the swarm, and the way the targets are analyzed from the standpoint of the swarm, conclusions are drawn about the expected information that the swarm collects with the sensors. This understanding comes from knowing how the swarm moves, and how those movements translate into distances and bearings, profile values and swarm densities. For this work, the concepts described in this chapter are implemented in a comprehensive software tool that allows the user to extract the information and display it in a manner that is easily understood.

4. Implementation and Results

This chapter presents the implementation software discussed in Chapter 3. Implementation of the information expectation project requires several pieces of software working together to create a tool that allows the user to make decisions based on visual and textual data presentation. The final product shows the user, in relatively rapid time, where to retrieved relevant information from the sensor data base with high confidence that the information is reliable enough to base decisions from.

This chapter describes the individual software components to facilitate understanding the project design and implementation. Section 4.1 describes the UAV swarm simulator used to generate swarm *track files*. Section 4.2 covers the implementation of the database used for the research. Section 4.3 details the design and strategies of the set up of the simulator, covering the set up of the UAVs and targets as well as the calculated values that are needed to determine the information expectation. Section 4.4 describes some of the tools available for querying the database and retrieving the information needed to determine information expectation. Section 4.5 discusses the various types of information, how that data is extracted from the database and what the algorithms and calculations are that bring the needed information to the user. Section 4.6 covers the methods developed to bring the user visualizations of the data in static and dynamic ways in both two and three dimensions.

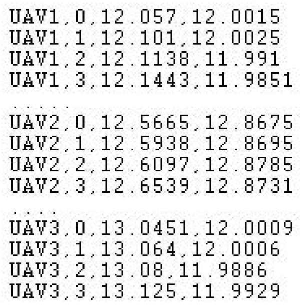
4.1 UAV Simulator

Because control of UAV swarm systems currently exist primarily as computational models, a swarm simulator produces the time-space data needed for this research. The swarm simulator used was developed at AFIT by Maj.Anthony Kadrovach [7]. Kadrovach's goal was to mimic the biological processes that enable swarming to take place in nature as seen in flocks of birds and herds of buffalo. The research was conducted to address four primary objectives:

1. The development of a swarm model with the flexibility in behaviors to be used in future swarm research
2. Analyze swarm behavior
3. Develop a classification methodology
4. Network development and analysis in the context of swarm behavior

This simulator uses the classic rules of separation, alignment and cohesion (Section 2.1.2) to compute individual agent positions. In addition, boundaries and way points are used to guide the swam as a whole. Boundaries limit the movement of the swarm to a defined rectangular region. Way points are stationary points that guide the swarm along a predetermined route while allowing the swarm members to maintain the swarm properties, within bounded time latency.

The output of the Kadrovach simulator is a file containing the positional data for the swarm elements. This is referred to as the *track file*. The positional data is stored in simple x-y format on a Cartesian coordinate system. Figure 4.1 shows the format of the track file. The format is follows: UAVid, Time, Latitude, Longitude. The speed and resolution of the simulation can be modified by changing the scale of the coordinate system and the time factor.



```
UAV1,0,12.057,12.0015
UAV1,1,12.101,12.0025
UAV1,2,12.1138,11.991
UAV1,3,12.1443,11.9851
.....
UAV2,0,12.5665,12.8675
UAV2,1,12.5938,12.8695
UAV2,2,12.6097,12.8785
UAV2,3,12.6539,12.8731
....
UAV3,0,13.0451,12.0009
UAV3,1,13.064,12.0006
UAV3,2,13.08,11.9886
UAV3,3,13.125,11.9929
```

Figure 4.1: Example Track File: UAVid, Time, Lat, Lon

4.2 Swarm Position Database Implementation

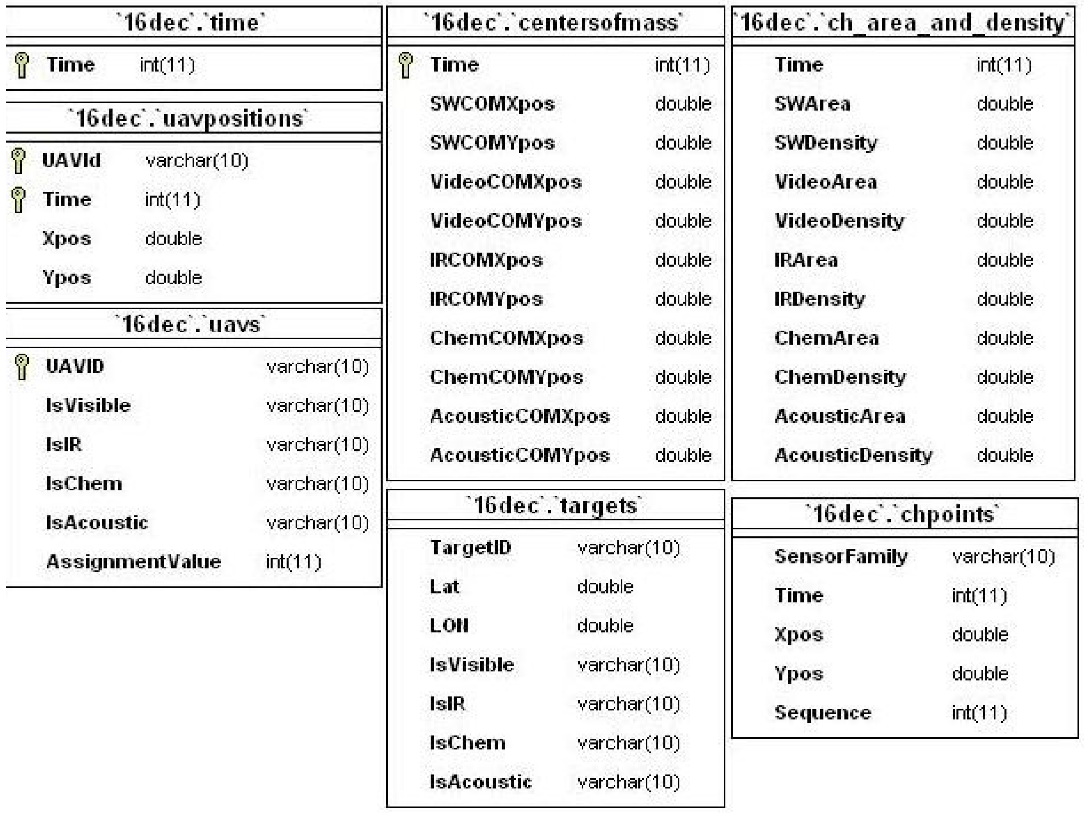
Modeling the data storage for the position data is done using a relational database. Using this type of storage allows for a structured approach to logging and retrieving the data while maintaining separate tables for targets, UAV platforms, sensors and time stamps.

The database used for this project is implemented using MySQL. This was chosen to allow the project to run on a variety of different platforms. The design of the database is straightforward and allows the extraction of several different types of information. Table 4.1 explains the various tables.

Table 4.1: Database Table Descriptions

Table	Description
TIME	Anchor table that is the basis for the simulation
UAVPOSITIONS	The database representation of the track file. All of the UAVs are generated from this table.
UAVS	Holds the information about each uav entity
TARGETS	Holds the information about each target entity
CENTERSOFMASS	Centers of mass for the whole swarm and the sensor family sub-swarms at each distinct time
CH_AREA_ AND_DENSITY	Convex Hull areas and sensor densities for the whole swarm and sensor family sub-swarms
CHPOINTS	The points that make up the convex hulls of the swarm and sub-swarms at each point in time

Figure 4.2 in shows the basic structure of the sensor database system. Information about the UAVs resides in the *UAVs* Table. This table has the identification of the UAV along with basic *TRUE* or *FALSE* for each type of sensor on board a sensor assignment value based on the assigned sensors is also stored. The time



\`16dec\`.\`time\`	
Time	int(11)

\`16dec\`.\`uavpositions\`	
UAVId	varchar(10)
Time	int(11)
Xpos	double
Ypos	double

\`16dec\`.\`uavs\`	
UAVID	varchar(10)
IsVisible	varchar(10)
IsIR	varchar(10)
IsChem	varchar(10)
IsAcoustic	varchar(10)
AssignmentValue	int(11)

\`16dec\`.\`centersofmass\`	
Time	int(11)
SWCOMXpos	double
SWCOMYpos	double
VideoCOMXpos	double
VideoCOMYpos	double
IRCOMXpos	double
IRCOMYpos	double
ChemCOMXpos	double
ChemCOMYpos	double
AcousticCOMXpos	double
AcousticCOMYpos	double

\`16dec\`.\`targets\`	
TargetID	varchar(10)
Lat	double
LON	double
IsVisible	varchar(10)
IsIR	varchar(10)
IsChem	varchar(10)
IsAcoustic	varchar(10)

\`16dec\`.\`ch_area_and_density\`	
Time	int(11)
SWArea	double
SWDensity	double
VideoArea	double
VideoDensity	double
IRArea	double
IRDensity	double
ChemArea	double
ChemDensity	double
AcousticArea	double
AcousticDensity	double

\`16dec\`.\`chpoints\`	
SensorFamily	varchar(10)
Time	int(11)
Xpos	double
Ypos	double
Sequence	int(11)

Figure 4.2: Database Model

stamps used throughout the simulation are in the *Time* table. Target information is stored in the *Targets* table. Each target is given an identification along with a position in latitude and longitude and boolean values that determine the sensors that can “see” the target.

In addition to the basic data taken from the swarm simulator output, the calculated data is also stored. The simulator calculates the convex hulls of the entire swarm as well as the sub-swarm sensor families at every discrete point in simulated time. The sets of points for the swarm and the sub swarms are sent to a convex hull class that runs the quickhull algorithm (Sec. 3.3) on the set of points to determine the points on the hull. This class also calculates the area of the convex hull. The calculated points and area are stored. The simulator also determines the location of the center of mass for the sensor families and stores that information in the database.

Storing this data is the purpose of the *centersofmass* tables. Finally, the coverage areas and sensor densities for the four sensor families are calculated and stored in the *CH_AREA_ AND_DENSITY* table.

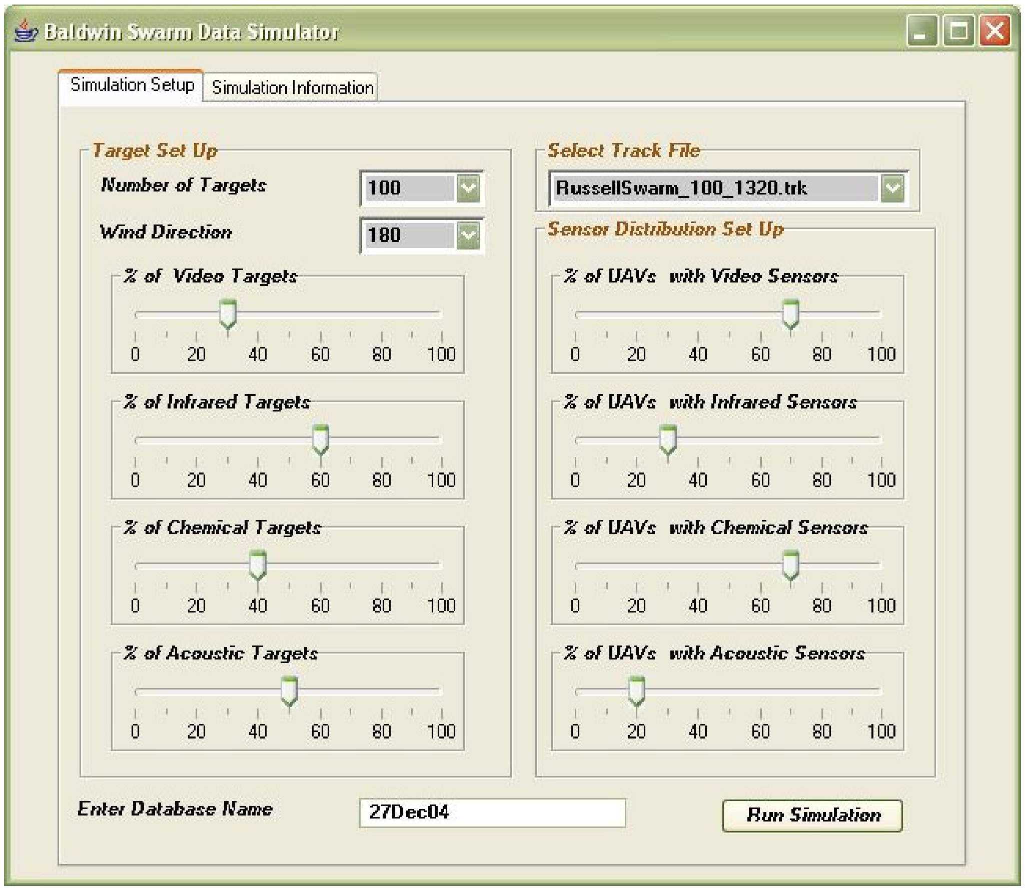
4.3 Swarm Position Analysis Tool

The Swarm Position Analysis Tool (SPAT) developed for this work enables the user to take a track file from a UAV simulator (or any other position in time simulator) and define a target field, sensor assignments and target type assignments. These assignments are placed in a database along with calculated values for several critical parameters of the swarm that are derived from the base track file. From the data placed in the database, the user is then able to extract numerous types of information that allow for a clear understanding of the times of expected sensor information from the swarm and sub-swarms about any target in the area of interest.

SPAT is developed using the Java programming language. This was chosen for ease of use, availability of third party components, and portability. MySQL$^{\text{TM}}$ is used for the database implementation, and PTPlot is used for graphing. Three dimensional visualization is accomplished using Audit Trail Viewer [2].

4.3.1 Simulation Setup. The Simulator Setup user interface (Fig. 4.3) is used to set the parameters for the analysis. The setup of the analysis begins with selecting a track file.

4.3.1.1 UAV Setup. The number of distinct UAV identifications pulled from the table created by the track file determines the number of UAVs in the analysis. A simple query outlined in Appendix A section A.1 returns the UAV identifications. The results of the query are scanned and each separate identification becomes a Java object. Once the number of UAVs is determined, the analyzer assigns sensors. These sensors are of one of four types: video, infrared, chemical or acoustic. Using the Simulator Setup user interface, the user selects the percentage of UAVs


Baldwin Swarm Data Simulator

Simulation Setup | Simulation Information

Target Set Up

Number of Targets: 100

Wind Direction: 180

Select Track File: RussellSwarm_100_1320.trk

Enter Database Name: 27Dec04

% of Video Targets
% of Infrared Targets
% of Chemical Targets
% of Acoustic Targets
Sensor Distribution Set Up
% of UAVs with Video Sensors
% of UAVs with Infrared Sensors
% of UAVs with Chemical Sensors
% of UAVs with Acoustic Sensors
0 20 40 60 80 100
Run Simulation

Figure 4.3: Simulation Setup User Interface

that are assigned each different type of sensor. If all of the sensor distribution setup sliders are set at 100%, every UAV is assigned every available sensor. If none of the sensor sliders are set at 100% there may be some UAVs that are not assigned any sensors. Once the analysis is started, sensors are assigned to UAVs in a random fashion using Algorithm 1.

Sensors assignments are accomplished by setting boolean variables in the object corresponding to the particular UAV to "TRUE." Upon assignment of the sensors, a sensor value is calculated and stored with the UAV object. This sensor value communicates the sensor assignments to visualization software. Section 4.6.3.1 outlines the details for this.

Algorithm 1 Sensor Assignment Algorithm

```
Set up empty "sensor" vector
NumberOfSensorsToAssign = NumberOfUAVs * Sensor assignment %
for NumberOfSensorsToAssign do
    tempValue=random # between 1 and NumberOfUAVs
    bool = FALSE
    if tempValue is in "sensor" vector then
        bool = TRUE
    end if
    if bool = FALSE then
        select UAV from TreeMap based on for loop key
        assign sensor to UAV
    else if bool = TRUE then
        decrement loop counter and get another random #
    end if
end for
```

Once the sensors are assigned to the UAVs, the database is populated with the characteristics of the UAVs. By storing these in the database, the system is able to identify certain UAVs used to calculate center of mass (Section 4.3.3). Table 4.2 shows the result of Algorithm 1 on a swarm of 32 UAVs with the default 50% sensor assignment selected for each sensor family.

4.3.1.2 Area Of Interest Setup. The area of interest for the analysis is set up as an area 10% larger than the area covered by the swarm. This is accomplished through the following algorithm:

Algorithm 2 Area Of Interest Setup

```
Select MaxLat, MinLat, MaxLon, MinLon from track file table
latBuffer = (MaxLat - MinLat) * .05
lonBuffer = (MaxLon - MinLon) * .05
MinLat = MinLat - latBuffer
MaxLat = MaxLat + latBuffer
MinLon = MinLon - lonBuffer
MaxLon = MaxLon + lonBuffer
Area of Interest = box bounded by MinLat, MaxLat, MinLon, MaxLon
```

The decision to make the area 10% larger than the area covered by the swarm is based on the need for targets to have varying ranges of coverage from the swarm. Creating the landscape larger than the area covered by the swarm durng its entire

Table 4.2: UAV Sensor Assignments

UAVID	IsVisible	IsIR	IsChem	IsAcoustic	AssignmentValue
UAV0	TRUE	TRUE		TRUE	13
UAV1	TRUE	TRUE	TRUE	TRUE	15
UAV2				TRUE	1
UAV3	TRUE	TRUE	TRUE	TRUE	15
UAV4	TRUE		TRUE		10
UAV5		TRUE			4
UAV6	TRUE	TRUE		TRUE	13
UAV7	TRUE	TRUE	TRUE		14
UAV8	TRUE		TRUE		10
UAV9					0
UAV10					0
UAV11		TRUE			4
UAV12			TRUE	TRUE	3
UAV13	TRUE	TRUE	TRUE		14
UAV14			TRUE	TRUE	3
UAV15			TRUE		2
UAV16	TRUE		TRUE	TRUE	11
UAV17				TRUE	1
UAV18		TRUE	TRUE		6
UAV19	TRUE	TRUE			12
UAV20		TRUE		TRUE	5
UAV21	TRUE				8
UAV22	TRUE	TRUE	TRUE		14
UAV23	TRUE	TRUE		TRUE	13
UAV24	TRUE	TRUE	TRUE		14
UAV25	TRUE			TRUE	9
UAV26					0
UAV27			TRUE		2
UAV28		TRUE		TRUE	5
UAV29	TRUE	TRUE	TRUE	TRUE	15
UAV30			TRUE	TRUE	3
UAV31				TRUE	1

flight ensures that there are some targets that receive no coverage from the swarm. The Java code used to accomplish this can be seen in Appendix B, Section B.2.2

4.3.1.3 Target Setup. Target setup is accomplished with the Simulator Setup user interface. The number of targets is selected from the drop-down box in the upper left of Figure 4.3. Any number from 1 - 100 in increments of 5 can be selected.

The first step to target placement is determining the bounds of the area of interest (section 4.3.1.2). Once that is accomplished, Algorithm 3 is used to randomly assign the targets within the area of interest.

Algorithm 3 Target Placement

```
1: Using final MinLat, MaxLat, MinLon and MaxLon values from Area Of Interest
   Algorithm Initialize four doubles:
2: lat = 0.0
3: lon = 0.0
4: LATRANGE = MaxLat - MinLat
5: LONRANGE = MaxLon - MinLon
6: for $i = 0$ to $numberoftargetsselected$ do
7:     lat = (random number between 0 and LATRANGE)+ MinLat
8:     lon = (random number between 0 and LONRANGE)+ MinLon
9:     name = "TGT" + i
10:    instantiate new Target object(name, point(lat, lon))
11: end for
```

The Java code used to accomplish this can be seen in Appendix B Section B.2.3. Figure 4.4 shows the results of the algorithm.

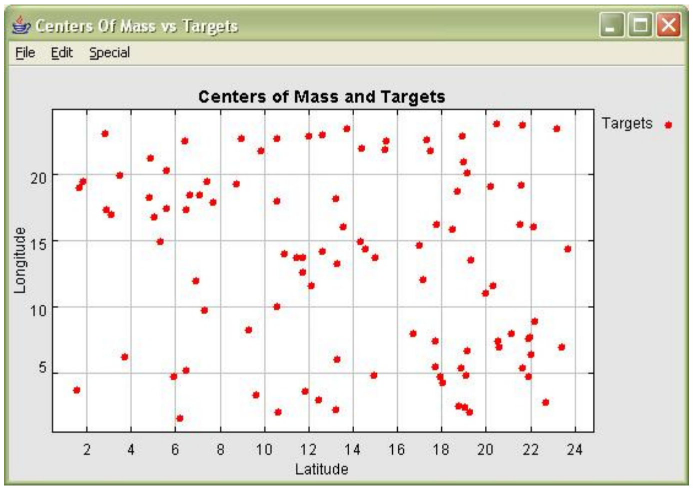


Figure 4.4: Random placement of targets over area of interest

Once the number of targets and their placement in the area of interest is determined, the user selects the percentage of targets that are visible by the various sensors. By selecting these percentages, the user has some control over the types of targets and the outcome of the analysis. Unless at least one of the percentage sliders is set to 100%, there exists the possibility that one or more of the targets is not assigned a target type, and thus is invisible to the sensors on board the UAVs. Target type assignments are done in a random fashion. Algorithm 4 determines the assignments. The Java code can be found in Appendix B Section B.2.4:

Algorithm 4 Target Assignment Algorithm

```
1: Set up empty "target" vector
2: NumberOfTargetsToAssign = NumberOfTargets * percentage
3: for NumberOfTargetsToAssign do
4:    tempValue=random # between 1 and NumberOfTargets
5:    bool = FALSE
6:    if tempValue is in "target" vector then
7:       bool = TRUE
8:    end if
9:    if bool = FALSE then
10:      select target from TreeMap based on for loop key
11:      assign phenomena to target
12:   else if bool = TRUE then
13:      decrement loop counter and get another random #
14:   end if
15: end for
```

4.3.2 Sensor and Target Modeling. For this work, the modeling of the sensors on the UAVs is limited to Boolean values that indicate whether a sensor is on board a particular UAV or not. Further modeling of sensor capabilities is discussed in section 6.1. At the setup of the simulation, a random number of UAVs are populated with sensors based on a user input, allowing wide flexibility in sensor composition scenarios. The mix of sensors is used to help determine the specific sensor-based information expectation on a target based on swarm movements. Section 6.1 details possible future work in the sensor modeling area that could be accomplished to make the simulator more realistic.

Targets are modeled in much the same way as sensors. Each target is assigned a set of phenomena that it "emits". These phenomena define the type of target. As with the sensors, the targets can be visible, infrared, chemical or acoustic targets. Each target may have all, some or none of these phenomena assigned, meaning the targets can each be of multiple types. There are several ways to model the targets that, along with the an accurate modeling of the sensors can create realistic numbers representing sensor readings in a sensor database. Section 6.2 outlines procedures which, if implemented, would yield a more realistic simulation environment in terms of having "sensor data" to analyze against expectation data.

4.3.3 Swarm Centers Of Mass. For this research, the centers of mass for the swarm as a whole and the sensor family sub-swarms are calculated and stored in the database. Algorithm 5 outlines the procedure.

Algorithm 5 Center of Mass Calculation Algorithm

```
 1: for each time slice do
 2:    calculate average lat for all swarm members
 3:    calculate average lon for all swarm members
 4:    calculate average lat for all swarm members with Video sensors
 5:    calculate average lon for all swarm members with Video sensors
 6:    calculate average lat for all swarm members with Infrared sensors
 7:    calculate average lon for all swarm members with Infrared sensors
 8:    calculate average lat for all swarm members with Chemical sensors
 9:    calculate average lon for all swarm members with Chemical sensors
10:    calculate average lat for all swarm members with Acoustic sensors
11:    calculate average lon for all swarm members with Acoustic sensors
12:    store calculated values in database
13: end for
14: order by time
```

The actual SQL statement is listed in Appendix A Section A.1. Inserting the data in the database makes it possible to easily retrieve the data and graphically depict the results. When this is done, the user is presented with a graph as seen in figure 4.5. This shows the path of the various swarms over the area of interest. With this information, the analyst begins to understand which targets are most likely to have been seen by the sensors, assuming the exact location of the targets is known as it is in this work.

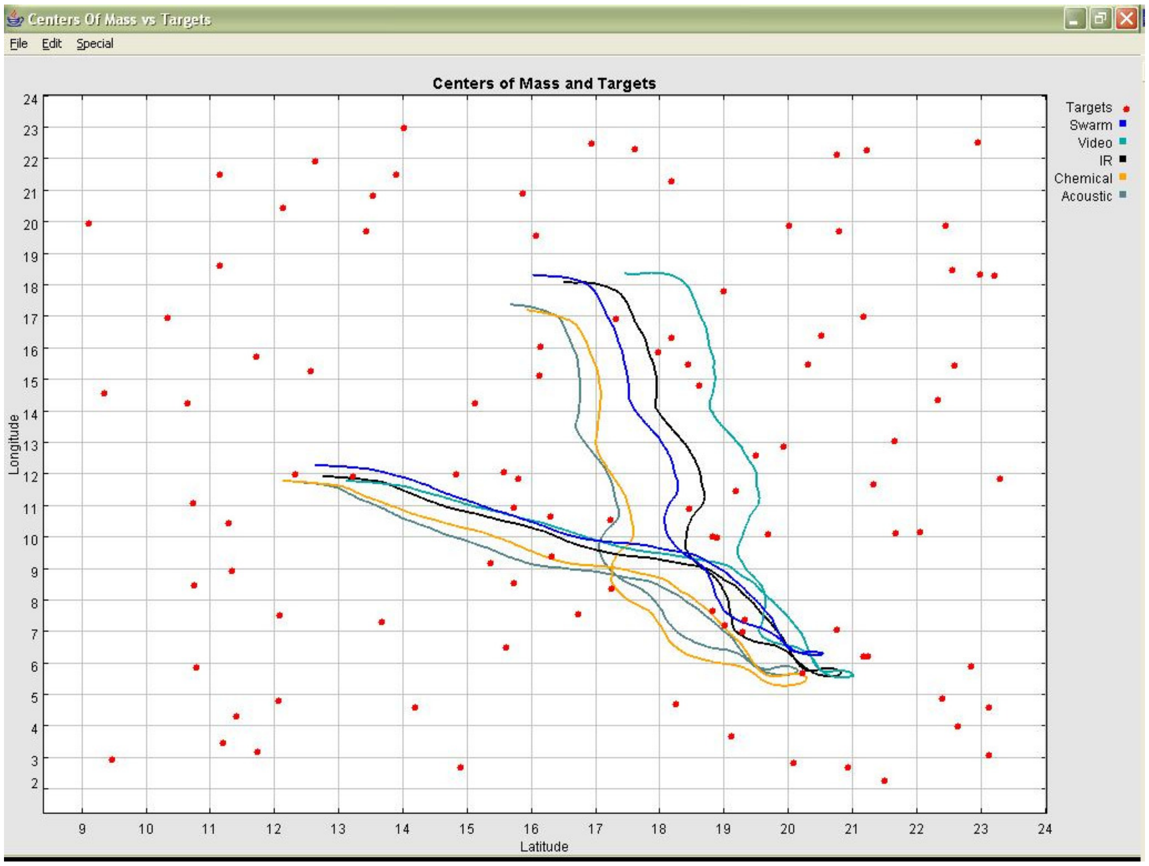


Figure 4.5: Plot of Centers of Mass for Swarms

4.4 Data Retrieval

Once the calculated data is stored, it is used to generate information expectation-related metrics. Several types of queries are executed against the data to extract information that conveys a level of expectation of relevant data. Table 4.3 lists example queries. To accomplish this, a GUI-based querying tool is created in Java.

4.5 Information Quality Expectation

The focus of this research is the development of methods that allow an analyst to determine when the best data is available from a sensor data store based on the movement of the swarm around a target field. In the following sections, the position data gathered during the swarm simulation, along with the derived data from the

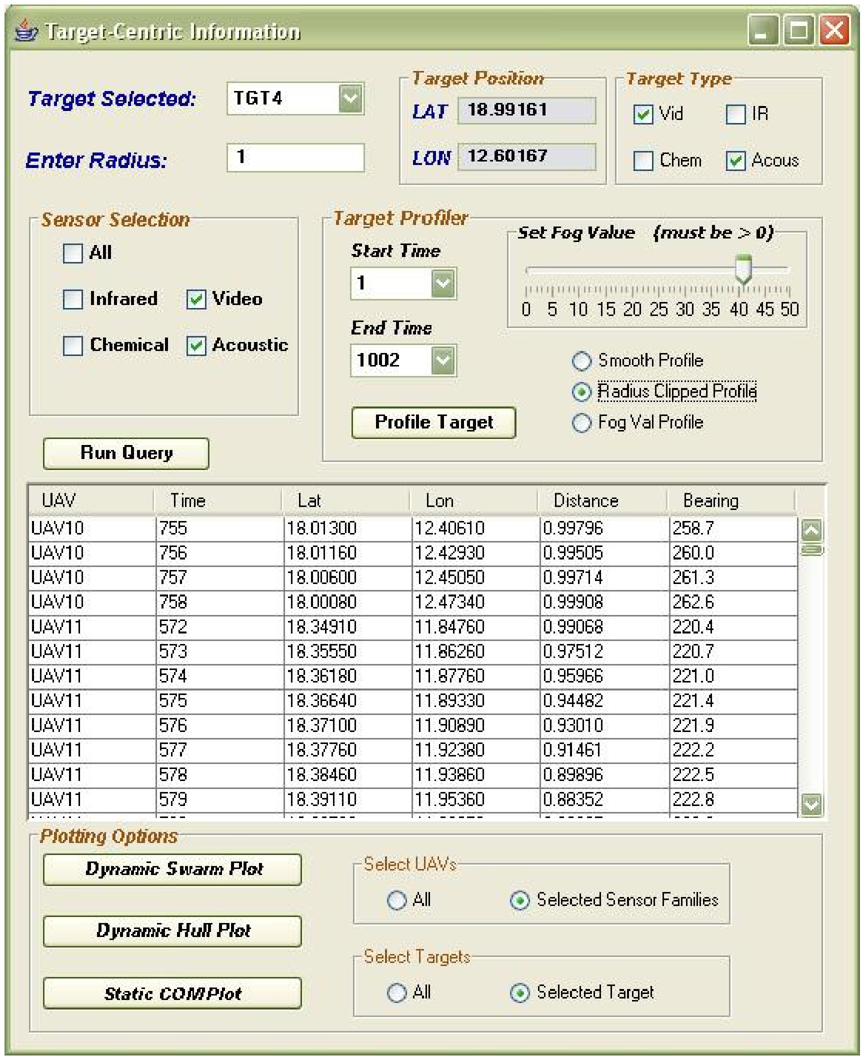
Table 4.3: Postprocessing Query Definitions

Query	Description
Eyes on target during a time period	Shows the complete list of sensors that should have data about a target during a specified period of time
Center of mass of a sensor family relative to a target	Shows the proximity of a sensor family's center of mass over time as compared to the location of a target
Sensor density during the period when the COM of a family was within a certain distance of a target	Shows how the coverage density changes over time. When coupled with the data showing how many of the sensors in the swarm saw the target, this gives insight into what can be expected given certain sensor densities.

analysis tool is retrieved from the database and presented to the user in various ways. These views each convey important information about what happened in the swarm at various times. It is possible to determine key times when a there is a high expectation of relevant sensor data present in the sensor data store through precise extraction and calculated manipulation of the data. Section 4.5.1 discusses the location-time data from the perspective of the targets, and what can be learned through querying the database from a target-centric point of view. Section 4.5.2 analyzes the data store from a sensor point of view and allows the user to see what is available when the sensors are the main focus of attention instead of the targets.

4.5.1 Target Centric Information Expectation. For the analysts using this tool, the reports about specific targets are the most useful. Figure 4.6 shows the user interface that controls the gathering of target information.

4.5.1.1 Target Profiling. Gathering this information requires that the database be queried for information on the target of interest. Using the Target Centric Information user interface (Fig. 4.6), the analyst selects a target from the list of available targets and a radius of interest is entered. This radius represents the area around the target that is of interest to the analyst. The target position is displayed



Target Selected: TGT4

Enter Radius: 1

Target Position

LAT 18.99161

LON 12.60167

Target Type

[✓ Vid] IR

Chem [✓ Acous]

Sensor Selection

All

Infrared [✓ Video]

Chemical [✓ Acoustic]

Target Profiler

Start Time 1

End Time 1002

Set Fog Value (must be > 0)

0 5 10 15 20 25 30 35 40 45 50

Smooth Profile [✓ Radius Clipped Profile] Fog Val Profile

UAV	Time	Lat	Lon	Distance	Bearing
UAV10	755	18.01300	12.40610	0.99796	258.7
UAV10	756	18.01160	12.42930	0.99505	260.0
UAV10	757	18.00600	12.45050	0.99714	261.3
UAV10	758	18.00080	12.47340	0.99908	262.6
UAV11	572	18.34910	11.84760	0.99068	220.4
UAV11	573	18.35550	11.86260	0.97512	220.7
UAV11	574	18.36180	11.87760	0.95966	221.0
UAV11	575	18.36640	11.89330	0.94482	221.4
UAV11	576	18.37100	11.90890	0.93010	221.9
UAV11	577	18.37760	11.92380	0.91461	222.2
UAV11	578	18.38460	11.93860	0.89896	222.5
UAV11	579	18.39110	11.95360	0.88352	222.8

Plotting Options

Select UAVs: All [✓ Selected Sensor Families]

Select Targets: All [✓ Selected Target]

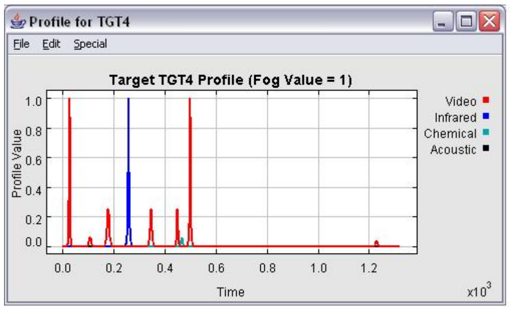
Figure 4.6: Target Centric Information user interface

along with type information (ie, the types of sensors to which it is "visible"). The user selects the fog tolerance value using the slider, and the start and end time for the profile. After setting these values, the database is queried for the UAVs and their boolean values that define the sensors on board. Once these are gathered, they are evaluated at each time slice as described in Section 4.5.1.1. Algorithm 6 steps through the procedure.

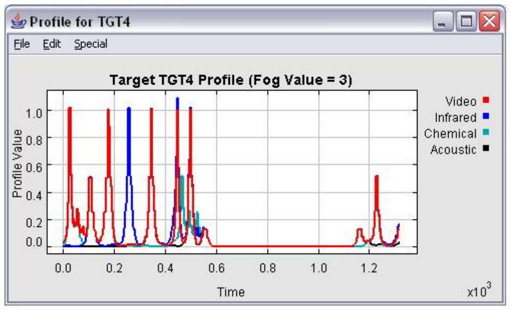
Algorithm 6 Target Profiling

```
1: Select all UAV data from database and put in “UAV“ collection
2: for $g = time_1$ to $time_n$ do
3:    set sensor profile values to 0
4:    while “UAV“ collection has more UAVs do
5:       Get boolean values for sensor assignments from UAV
6:       Calculate distance from UAV to target
7:       $exp = \frac{distance}{foginterval}$
8:       $Value = \frac{1}{2.0^{exp}}$
9:       for all types of sensors do
10:         if UAV has sensor on board then
11:            Value is added to Sensor Profile Value
12:         end if
13:      end for
14:   end while
15: end for
```

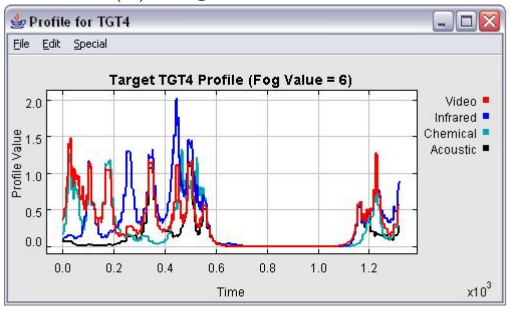
The resulting set of values represent a time profile of the availability of information on a certain target. The higher the value relative to other values in the profile, the higher the confidence that there will be information available about the target. Figure 4.7 shows examples of the various profiles on a single target. The six profiles show the difference between high and low fog tolerance values. When the fog tolerances are high, the profile values are inflated, and the perception of the amount of relevant information is correspondingly inflated. In reality, the lower fog tolerance value profiles pinpoint the times where the highest quality information was actually taken. Figure 3.4 (page 30) brings to light the reason why. As the concentric rings increase in width, higher values extend further out from the target, thus being assigned to more UAVs. As the tolerance value decreases, the rings get smaller. In this case, only when the UAVs pass close to the target are they assigned relatively high values. An important point is to note that the actual profile values themselves are not as important as the combination of the profile values and the fog tolerance value. The combination of a low fog value and a relatively high profile value establishes the highest expectation of relevant data.



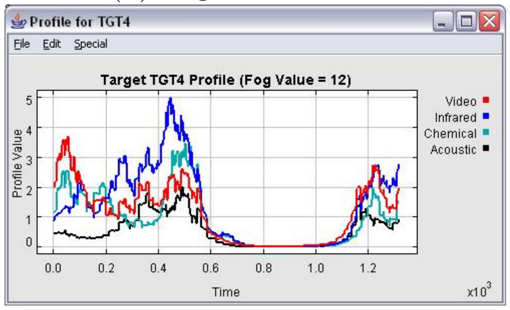
(a) Fog tolerance = 1



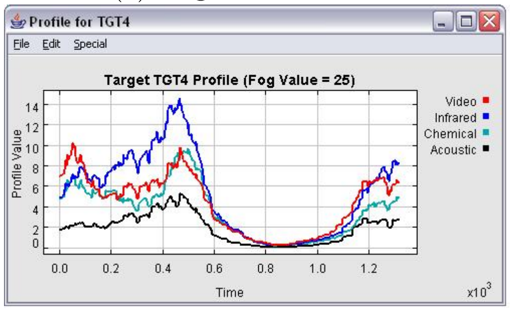
(b) Fog tolerance = 3



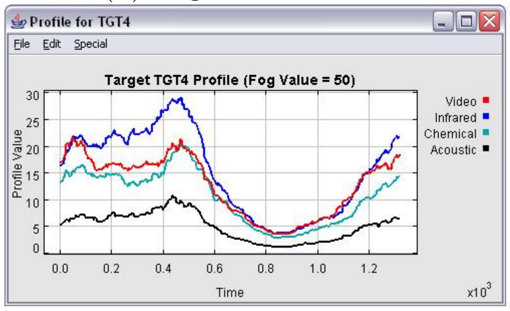
(c) Fog tolerance = 6



(d) Fog tolerance = 12



(e) Fog tolerance = 25



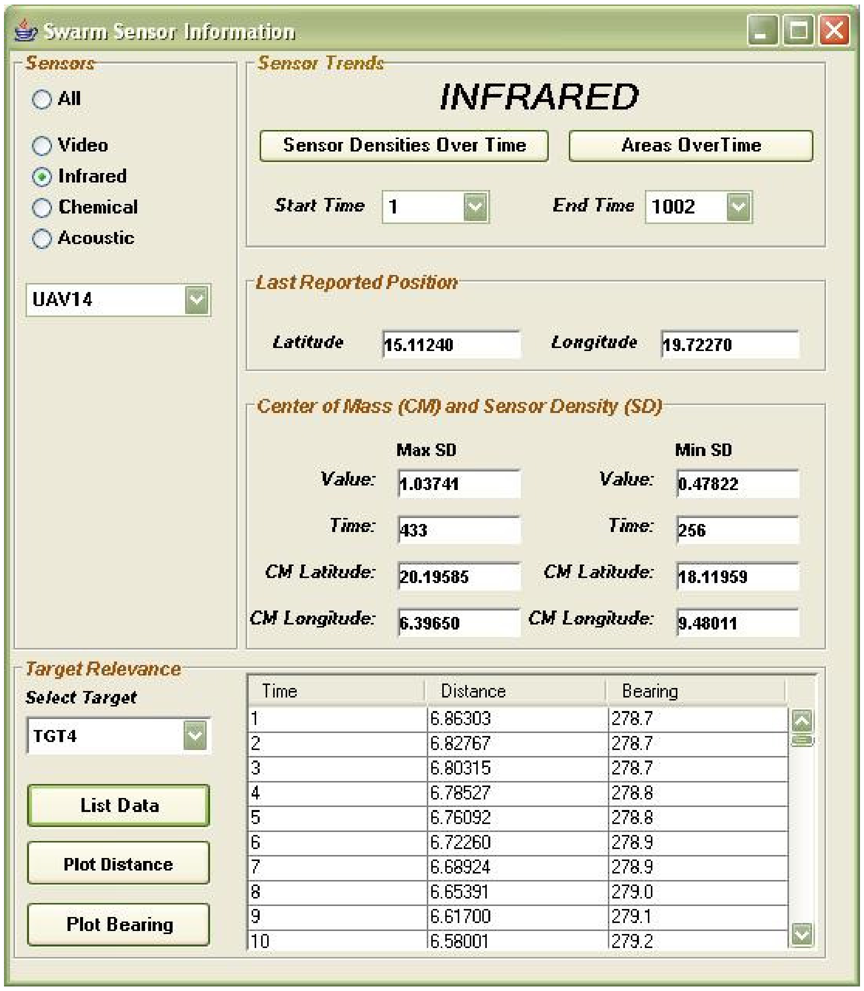
(f) Fog tolerance = 50

Figure 4.7: Profile plots with varying fog tolerance values

4.5.1.2 Target-Sensor Proximity. By looking at the results of a target profile, an analyst draws conclusions about the likely availability of relevant information from the swarm relative to a time-location of interest. Spikes in the plot indicate that one or more swarm members with the appropriate sensor passed within a distance appropriate for data collection. What the profile doesn't show is the identification of the UAV or set of UAVs who's close proximity to the target created the spike in the plot. The target-sensor proximity query tool is used to identify the exact swarm members that created the increase in expected information. Using the Target Centric Information user interface (Fig. 4.6), the user selects the target of interest, the distance within which the returned UAVs must have passed, and the sensor or sensor types that must have been on board for the UAV to be considered as providing relevant information. The result is the table in Figure 4.6 that lists the UAVs that came within the prescribed distance, the times when they were within range, their geographic position at the times, as well as their distance and bearing from the target.

4.5.1.3 Possible Data Query. The results of the previous query narrows the search for data from the sensor database to a relatively small area. The previous query returns the UAVs that passed within a distance where relevant data is likely, as well as the times the passes occurred. The analysts now begin to formulate queries to the swarm sensor data store that would return data of reasonable relevance.

4.5.2 Sensor Centric Results. By looking at the swarm from a sensor-centric point of view, the analyst sees what resources are available, and gets an overall look at the swarm composition and the coverage availability over time. The analyst uses the the Swarm Sensor Information user interface (Fig. 4.8) to set up queries to extract these data. The user selects the swarm family from the list. Either the entire swarm or specific sensor families can be selected. Once the needed swarm is selected, the drop down box is populated with the swarm members. The center of mass for the swarm is also loaded. By selecting a start and end time, the user sees the areas of coverage (Sec. 4.5.2.1) or the sensor densities (Sec. 4.5.2.2) over time. The last reported position of the selected swarm entity (center of mass or individual swarm member) is displayed. The interface also gives the user a quick look at when the maximum and minimum sensor density occurred and the location of the center of mass of the swarm at that time.



Sensors: All, Video, [✓ Infrared], Chemical, Acoustic

UAV14

Sensor Trends

INFRARED

Start Time: 1 End Time: 1002

Last Reported Position

Latitude: 15.11240 Longitude: 19.72270

Center of Mass (CM) and Sensor Density (SD)

	Max SD		Min SD
Value:	1.03741	Value:	0.47822
Time:	433	Time:	256
CM Latitude:	20.19585	CM Latitude:	18.11959
CM Longitude:	6.39650	CM Longitude:	9.48011

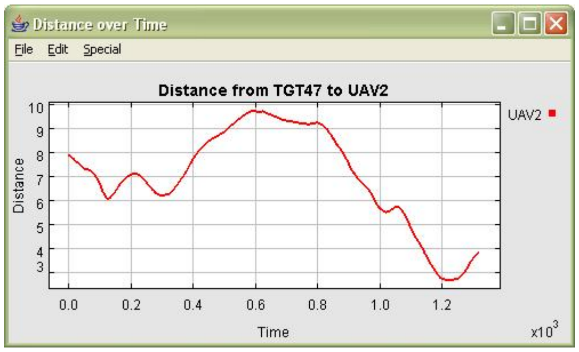
Target Relevance

Select Target: TGT4

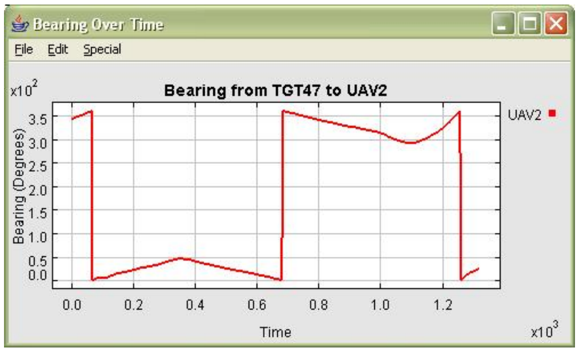
Time	Distance	Bearing
1	6.86303	278.7
2	6.82767	278.7
3	6.80315	278.7
4	6.78527	278.8
5	6.76092	278.8
6	6.72260	278.9
7	6.68924	278.9
8	6.65391	279.0
9	6.61700	279.1
10	6.58001	279.2

Figure 4.8: Swarm Sensor Information User Interface

The user selects a target from the drop down list, and selects the “List Data” button to see the proximity of the UAV to the target throughout the data run. The “Plot Distance” and “Plot Bearing” buttons present the user with plots depicting the information in graphical form. Figure 4.9(a) shows the distance from the target that the UAV was over the entire data run. At a glance it becomes evident that the distance was shortest at the end of the run. Figure 4.9(b) shows the bearing from the target to the UAV. The sudden jumps in the plot indicate that the UAV was due north of the target and crossed from northwest to northeast of the target and back a couple of times during the data run. These plots obviate the need to scan the tabular data for time references and distances. From this the user sees, at a glance, the UAV's position in time compared to the target. The information from these assist the user in determining the best time to ask for information from the swarm sensor data store.



(a) Distance over time

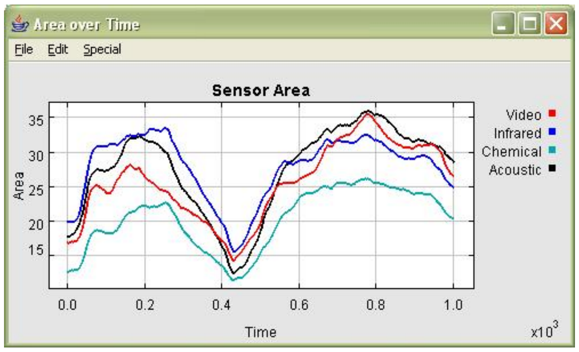


(b) Bearing over time

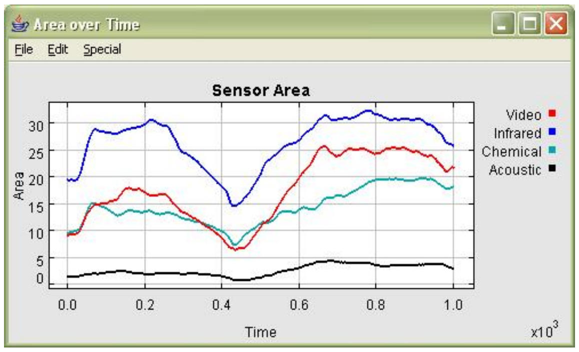
Figure 4.9: Plots of distance and bearing over time

4.5.2.1 Areas Of Coverage. As the swarm moves around the area of interest, the dynamic nature of the swarm causes the area of coverage to change with time. This is true for the entire swarm as well as the individual sensor family sub-swarms. As these areas change, the probability of detecting targets changes as well.

Figure 4.10 shows the result of querying the database for the areas of four sensor families for the entire data run. Of particular interest is the shape of the plots. It is reasonable to expect that the lines will generally follow the same pattern. As the swarm moves, the swarm control algorithms will tend to grow and shrink the coupling of the members with respect to each other. Not surprising, the areas of coverage for the sensor families are relatively similar in the case of Fig. 4.10(a) where the number of members in each family of sensors is similar. This changes if the percentage of UAVs with particular sensors is different for the various sensor families (Fig. 4.10(b)).



(a) Identical sensor family sizes



(b) Varying sensor family sizes

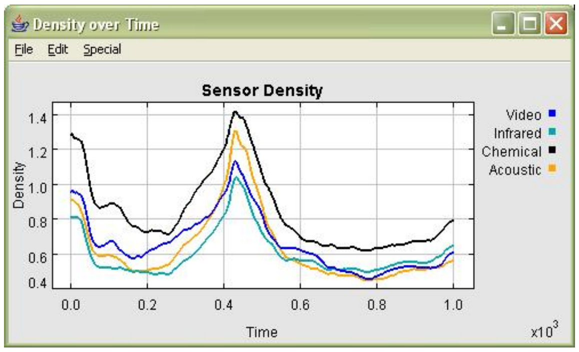
Figure 4.10: Plots of areas of coverage

4.5.2.2 Sensor Density. Figure 4.11 shows the change in sensor density over time. As with the area plots (Fig. 4.10), a change in the number of sensors has an effect on the change in sensor density. As mentioned in Section 3.3.2, a change in sensor density speaks to a change in the expected corroborability of data from members of the sensor family. As the density increases, the likelihood of multiple sensors seeing a target simultaneously increases. Simultaneous target data allows analysts to cross reference the these data to validate sensor readings.

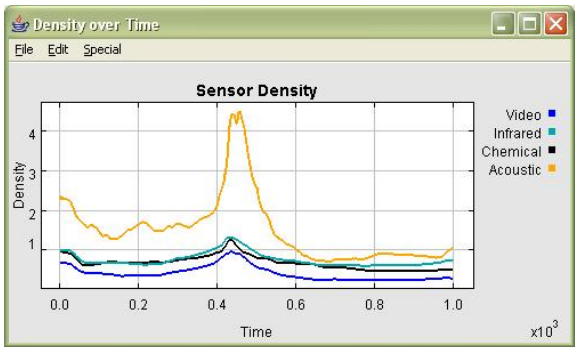
4.6 Data Visualization

In this work, the ability to understand swarm behavior is given to the user through several static and dynamic plots. These plots present information that facilitates understanding the swarm's coverage of the target field.

A set of Java classes designed specifically for plotting are integrated into the software written for this project. PTPlot is a plotting package from the Ptolemy II project originating at the University of California at Berkely [3]. Ptolemy II is a set of Java packages that support heterogeneous, concurrent modeling and design.



(a) Identical sensor family sizes



(b) Varying sensor family sizes

Figure 4.11: Plots of sensor densities

PTPlot is a plotting application programming interface (API) that is part of the project. This API was used to generate the static plots and the two dimensional dynamic plots discussed in sections 4.6.1 and 4.6.2 respectively.

In addition to two dimensional static and dynamic plots, three dimensional swarm visualization is integrated into this research. For this aspect of the project, JView is used. JView is a Java visualization API from Air Force Research Laboratories in Rome, NY [2]. This software allows visualization of data through its Audit Trail Viewer (ATV) product. By using ATV with some modifications (see section 4.6.3.1), the user is able to see the swarm movements from several different angles.

4.6.1 Static Plots. Several immediate-information plots are available to the user. These plots represent such metrics as distance and bearing between selected targets and UAVs, sensor densities over time and coverage areas over time. Table 4.4 lists the plots available and what they represent.

4.6.2 Dynamic Plots. Visualizing the swarm in motion presents a unique challenge. Displaying a sequence of swarm positions in chronological order is possible because the position data stored in the database is attributed to time. Two specific

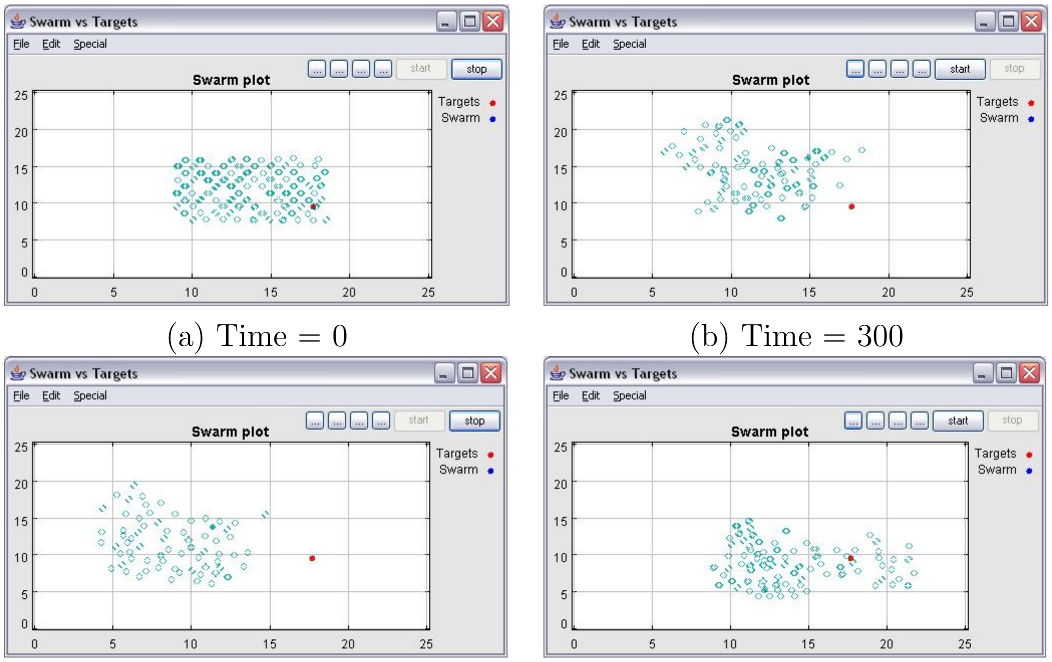
Table 4.4: Static Plots

Plot Title	Description	Example Fig
Centers of Mass vs Targets	Position of the centers of mass of the sensor families with regards to the targets	4.5
Distance Over Time	Distance between a selected target and a selected UAV over the duration of the simulation	4.9(a)
Bearing Over Time	Bearing from a target to a UAV for the duration of the simulation	4.9(b)
Areas Over Time	Size of the areas covered by of the sensor families over a selected time interval	4.10(a) and 4.10(b)
Density Over Time	Densities of the sensor families over a selected time interval	4.11(a) and 4.11(b)

dynamic plots are available that give the user a "birds eye" view of the swarm over the area of interest.

Figure 4.12 shows the swarm moving over the targets at four distinct times. These plots allow the user to see exactly where the swarm moved throughout its flight. The user can select whether to view the entire target and UAV field, or select from individual targets and/or individual UAV sensor families. In this case, a single target is selected with the entire swarm moving in the area of interest

The second plot (Fig. 4.13) shows the user the convex hull of the sensor families over the targets. In this particular example, the outlines represent the infrared sensor family (outer polygon) and the acoustic sensor family (inner polygon). From this plot one can see that the infrared sensors have a larger area of coverage than the acoustic sensors at this particular time in the data run. As this plot runs, the convex hull representing the area of coverage for particular sensor families moves over the target area. With the ability to select individual targets, this plot gives the user immediate feedback as to the possibility of coverage for a particular target. If the



(a) Time = 0 (b) Time = 300

(c) Time = 800 (d) Time = 1300

Figure 4.12: Swarm over target at varying times

target never enters an area enclosed by a convex hull, there will be no data for that target. Once the units of measure are determined, this plot also allows the user to see how close the sensor coverage came to the target.

Both of these plots are created by implementing a Dynamic Plot class in the software. Dynamic Plot is a class derived from the PlotLive class in the PTPlot software. The class defines a set of static points (targets) that are always visible on the screen. These points are held on the plot by a variable that defines a base set of points to display continuously. The software queries the database for the positions of the swarm members in chronological order and persists a display of the points from a set time for a pre-determined duration. As the new points are brought in from the database, the old points are disposed of and the new points are displayed. The effect of this is a moving plot that shows the progression of the swarm over the area of interest.

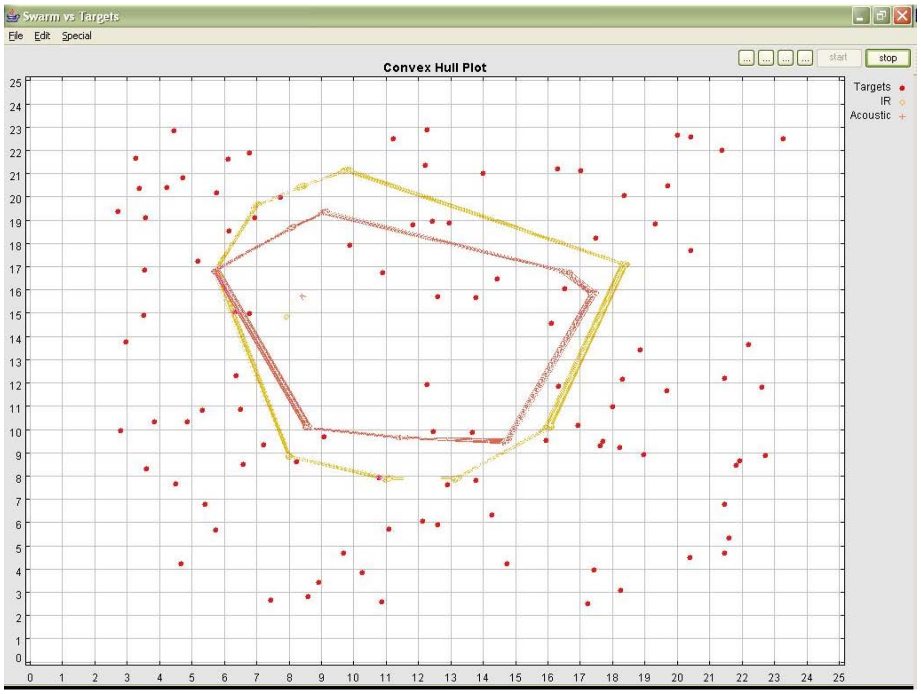


Figure 4.13: Convex Hulls of Sensor Families Over Targets

4.6.3 Audit Trail Viewer. Audit Trail Viewer provides a way of looking at the swarm and/or the targets in three dimensions, as well as animating the movements of the swarm. This product was developed by the Air Force Research Laboratory's information directorate (AFRL/IF) in Rome, NY. The purpose of the tool is to visualize movements based on tracking data input into the system. In the case of this research, the data is drawn from the database and a specialized file is created to visualize the area of interest and the swarms.

Visualizing the swarm in three dimensions is an important part of information expectation. The ability to grasp events from multiple perspectives sheds light on anomalies in data, and helps explain differences in sensor readings. ATV allows the user to "fly along" with the swarm and see what is happening from the perspective of any of the targets, UAVs or from anywhere around the area of interest. With a few modifications to the code, the ATV software has the ability to visualize the convex hulls of the various sensor families as well.

The following sections detail the modifications to the software, the creation of a specialized file type and the results of the ATV software and how it is used to visualize swarm events.

4.6.3.1 Code Modification. The modifications to the ATV code involve creating a new section of the user interface to allow the user to select what sensor family convex hull would be shown. The user interface uses basic Java Swing components and represents a small part of the modification. The underlying code that generates the convex hulls is implemented from scratch. This Java class retrieves the sensor value (Sec 4.6.3.2) from the individual UAVs and determines the sensors on board from that number. Four Java containers are created, one for each sensor family, that hold the identity of the sensor family members. At each time slice the locations of the points for each family are sent to a convex hull class that determines the hulls for the individual families. That information is sent to the graphics package and the hull is drawn on the screen.

4.6.3.2 Sensor Values. Each UAV is assigned a sensor value based on the sensors assigned to that UAV. Video sensors have a value of 8, IR sensors 4, chemical sensors 2, and acoustic sensors 1. The sensor value for the UAV is the sum of the individual sensor values of the sensors assigned to the UAV. Table 4.5 shows the concept. By using this scheme, a representation of the sensors on board is a single integer instead of a collection of four boolean variables. Software needing to decipher what sensors are on board create a four bit binary word from this integer and look at the bits set to “1” to determine the assigned sensors.

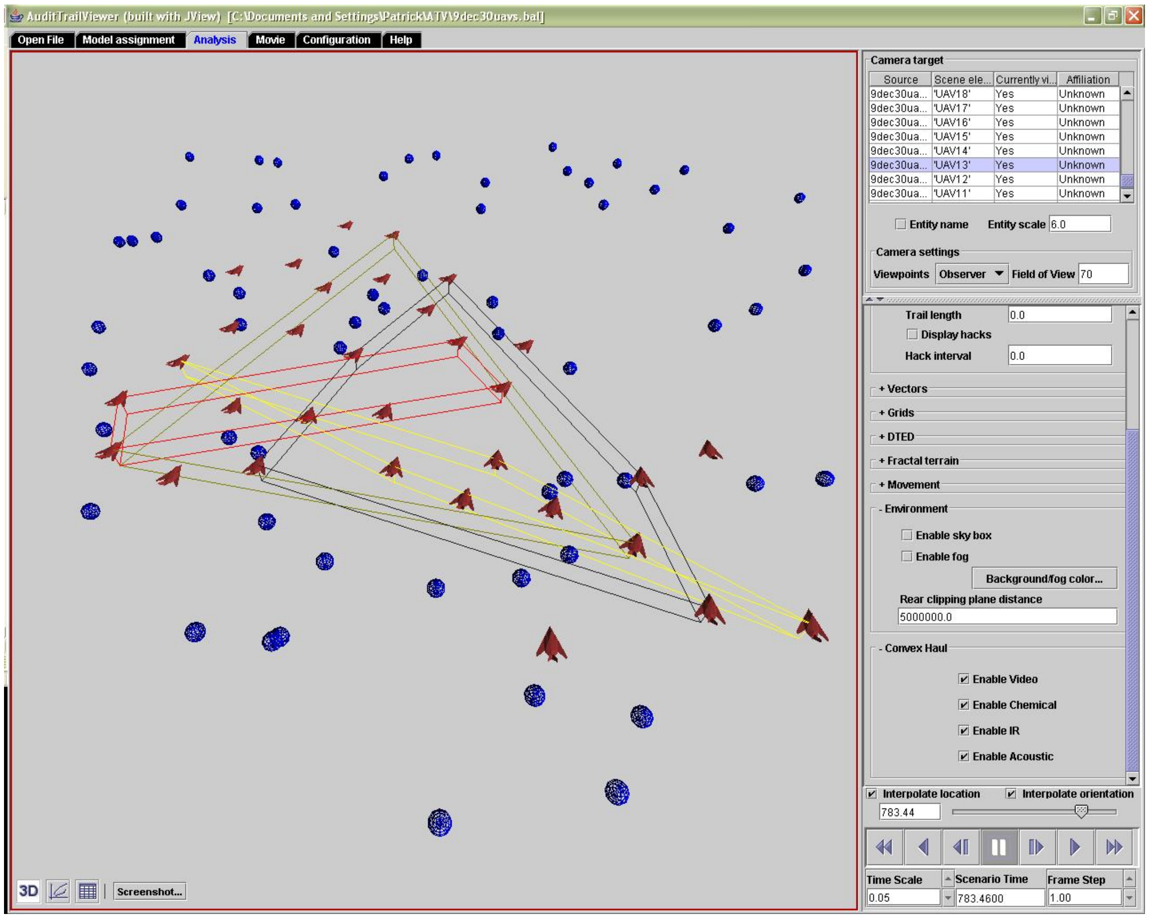
4.6.3.3 ATV File. The base data file used by ATV consists of a declaration of the entities shown on the screen along with the time-positional data for the motion of the entities. For each entity at each time slice, the latitude, longitude, altitude, roll, pitch, and yaw of the entity are provided. Building the ATV file

Table 4.5: UAV Sensor Value Assignments

UAVID	Video (8)	IR (4)	Chem (2)	Acoustic (1)	Sensor Value
UAV0	1	1	0	1	13
UAV1	1	1	1	1	15
UAV2	0	0	0	1	1
UAV3	1	1	0	0	12
UAV4	1	0	1	0	10
UAV5	0	1	0	0	4

requires a modification to the base data file that ATV normally uses. Latitude and longitude are set by the swarm agents' X-Y position stored in the database. Altitude is set to a constant value determined by the user. Roll and pitch do not influence the visualization of the swarm and as such are set to zero throughout the simulation. In a real swarm, these would obviously affect the platform's look down angle. Yaw angle reflects heading and is easy to calculate and include into the ATV data file. This enables ATV to accurately portray the heading of the aircraft as they move through the simulation. This is important because it allows the user to see how the individual swarm agents interacts with the one another. Another significant change is the addition of the sensor values (Sec. 4.6.3.2) to the declaration of the entities to calculate the convex hulls (Sec. 4.6.3.1).

4.6.3.4 ATV Results. The Audit Trail Viewer is the key that enables information expectation visualization. The following screen shots depict the type of information that can be seen from the ATV simulations. Each of these screen shots is captured from a single simulation run frozen at a point in time. The altitude of the UAVs is set artificially low to capture the effect of the the UAV swarm moving over the targets.



Camera target

Source	Scene ele...	Currently vi...	Affiliation
9dec30ua...	'UAV18'	Yes	Unknown
9dec30ua...	'UAV17'	Yes	Unknown
9dec30ua...	'UAV16'	Yes	Unknown
9dec30ua...	'UAV15'	Yes	Unknown
9dec30ua...	'UAV14'	Yes	Unknown
9dec30ua...	'UAV13'	Yes	Unknown
9dec30ua...	'UAV12'	Yes	Unknown
9dec30ua...	'UAV11'	Yes	Unknown

☐ Entity name Entity scale 6.0

Camera settings
Viewpoints Observer Field of View 70

Trail length 0.0
☐ Display hacks
Hack interval 0.0

☐ Enable sky box
☐ Enable fog
Rear clipping plane distance 5000000.0

✓ Enable Video
✓ Enable Chemical
✓ Enable IR
✓ Enable Acoustic

✓ Interpolate location ✓ Interpolate orientation
783.44

Time Scale	Scenario Time	Frame Step
0.05	783.4600	1.00

Figure 4.14: Full ATV GUI showing the swarm as it moves over targets.

Figure 4.14 displays the swarm visualizer interface. On the right are the controls for selecting the camera placement, entity scale, viewpoints, trails, vectors, grids, DTED, terrain, movement , environment and convex hull. Each of these allows ATV to either show different types of information about the simulation or enhance the simulation by overlaying environmental data onto the display. For this work, the convex hull tools were the focal point. In all of the figures, the convex hulls can be seen as the lines between the various UAVs forming double polygons. The four polygons drawn from the UAVs to the ground are the convex hulls of the different sensor families.

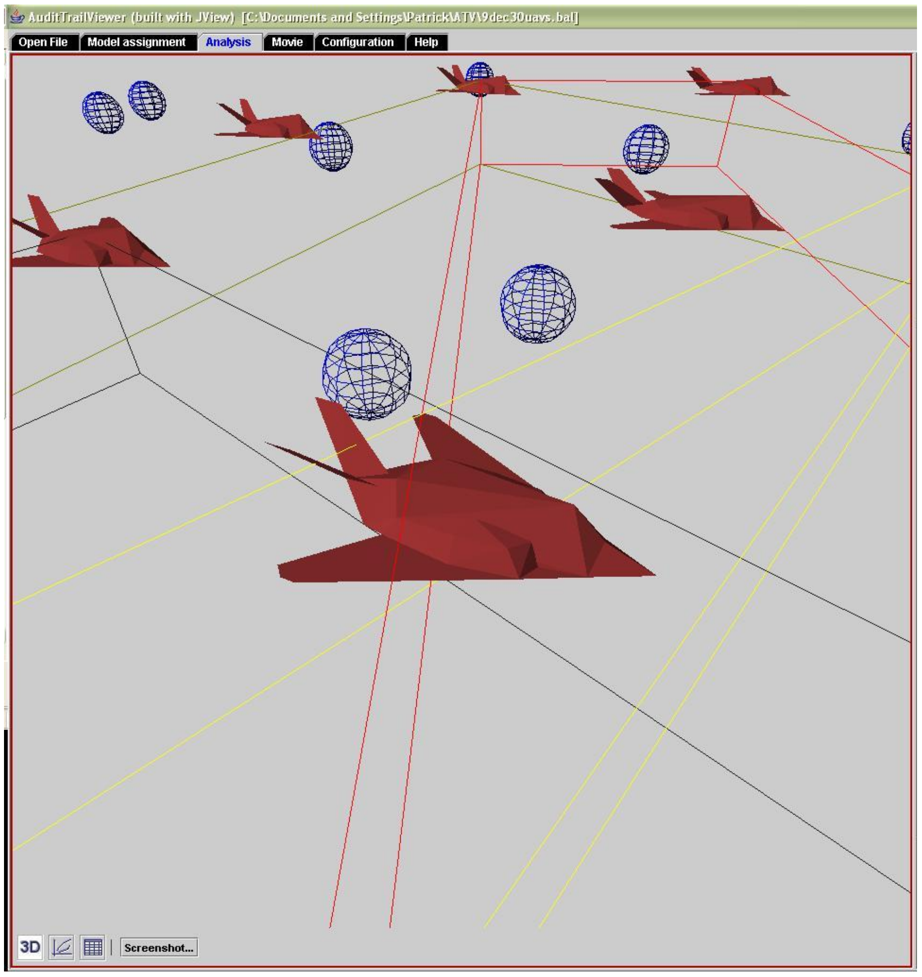

AuditTrailViewer (built with JView) [C:\Documents and Settings\Patrick\ATV\9dec30uavs.bal]
Open File
Model assignment
Analysis
Movie
Configuration
Help
3D
Screenshot...

Figure 4.15: Close up view of six UAVs overflying targets (spheres)

Figure 4.15 shows a zoomed in portion of the swarm as it moves over the target field. By using one UAV as the camera focal point, the user can rotate the view from any angle and zoom in and out as the visualization is in motion.

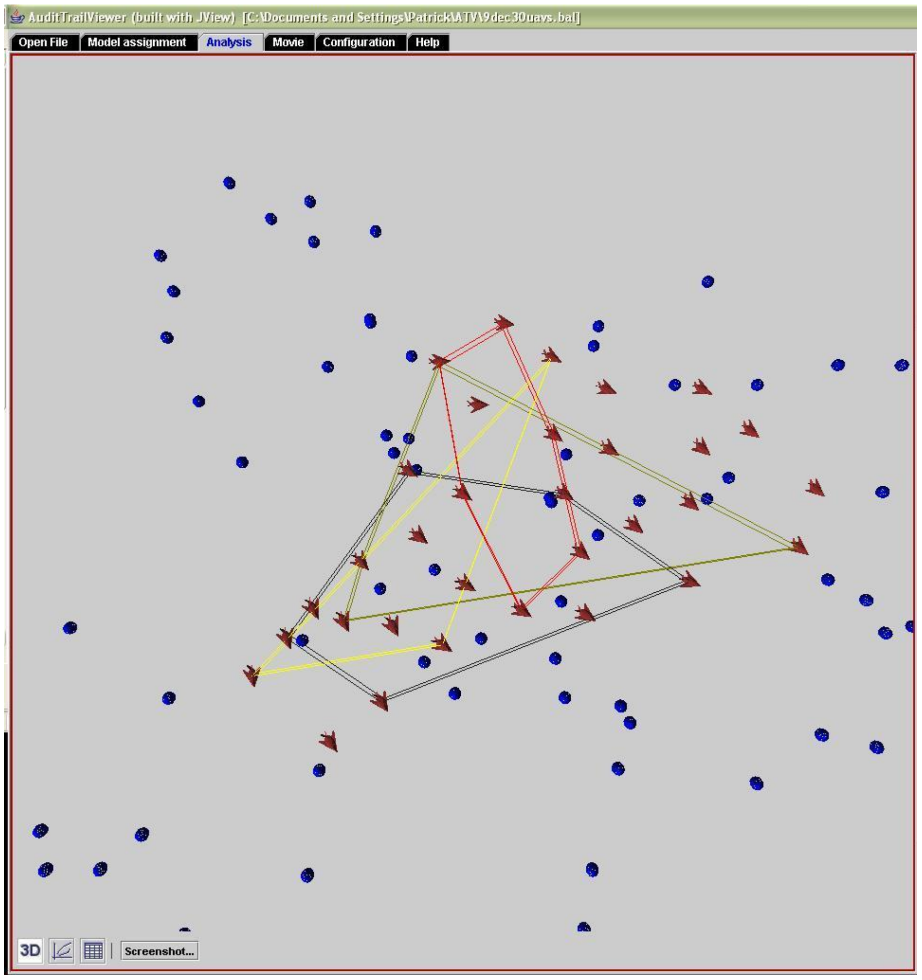

AuditTrailViewer (built with JView) [C:\Documents and Settings\Patrick\ATV\9dec30uavs.bal]
Open File
Model assignment
Analysis
Movie
Configuration
Help
3D
Screenshot...

Figure 4.16: Zoomed out view of the swarm over an area of interest

Using the Bird's Eye view (Fig. 4.16) its possible to get an overall view of the area of interest. The user can toggle the entity names on and off as needed to get instant feedback about which UAVs are near which targets at any time.

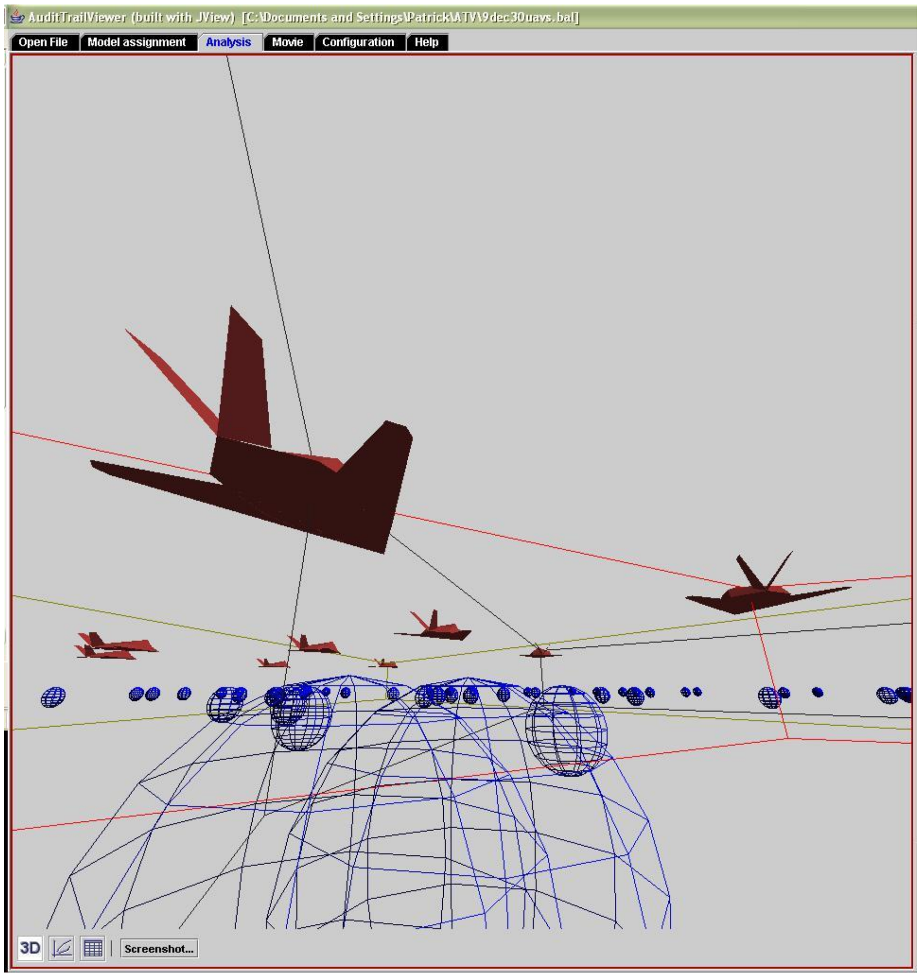

AuditTrailViewer (built with JView) [C:\Documents and Settings\Patrick\ATV\9dec30uavs.bal]
Open File
Model assignment
Analysis
Movie
Configuration
Help
3D
Screenshot...

Figure 4.17: View from near a target looking up at the swarm as it passes overhead

Figure 4.17 illustrates the capability of looking at the swarm from a target-centric point of view. From this view, the user can look in every direction and see the swarm as it moves around the target. This allows the user to visualize the coverage of the target over the duration of the swarm flight.

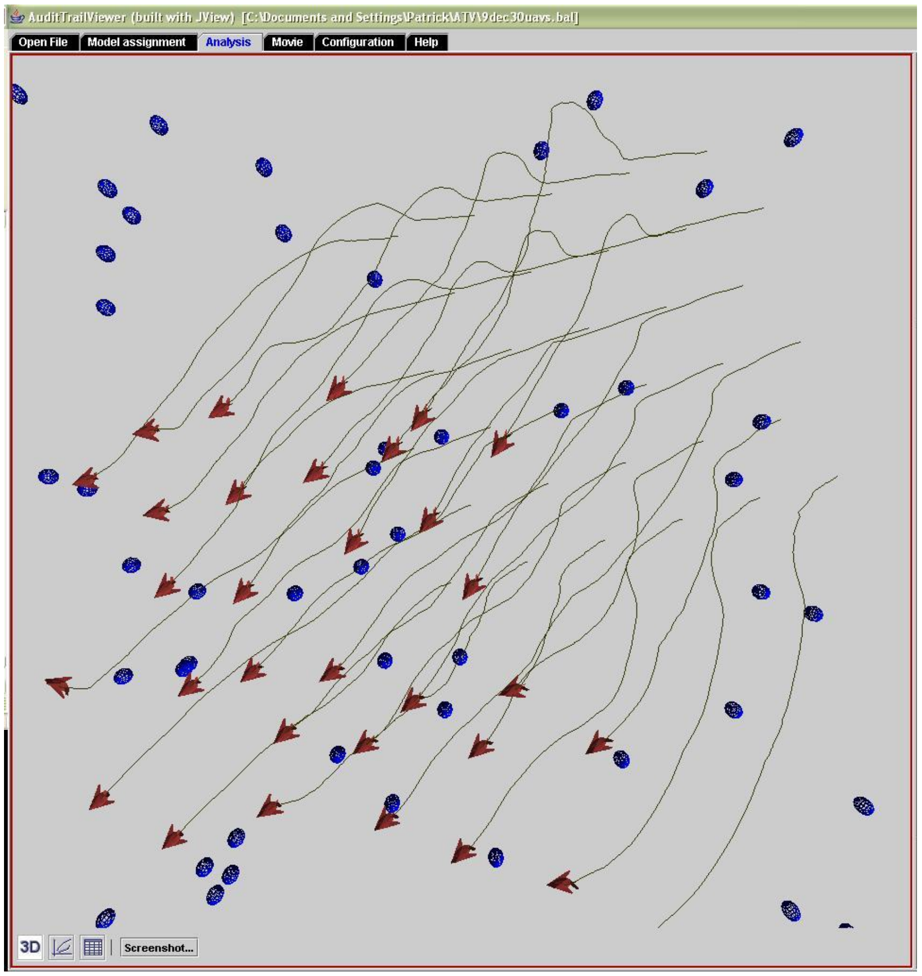

AuditTrailViewer (built with JView) [C:\Documents and Settings\Patrick\ATV\9dec30uavs.bal]
Open File
Model assignment
Analysis
Movie
Configuration
Help
3D
Screenshot...

Figure 4.18: Bird's Eye view of swarm member routes. Convex hulls are turned off

Figure 4.18 shows ATV's ability to visualize the past paths of the swarm. The lines show the paths swarm members took and allow a user to visualize which targets specific members are likely to have data on. Knowing this allows the user to retrieve sensor data from one UAV and compare sensor readings from various targets. ATV also provides a look ahead route preview option.

4.7 Summary

This chapter provides the implementation methodology for the analysis model developed in Chapter 3. A swarm simulator is introduced as the source for positional data. As a means of storing that data as well as derived data, a swarm position database is developed. The database works with the Swarm Position Analysis Tool (SPAT), also developed in this work. Information quality expectation derivation is implemented through both target-centric and sensor-centric information retrieval methods implemented in the SPAT. Finally, several visualization techniques are created that allow the analyst to view static and dynamic information using a combination of 2-dimensional static and dynamic plots as well as a 3-dimensional visualization of the swarm. These 3 dimensional visualizations enable the user to see the swarm in motion from the point of view of any swarm member or target.

5. Conclusions

As stated in Chapter 1, the focus of this work is developing methods and algorithms for assessing the degree of coverage of UAV swarm elements over a set of distinct points in an area of interest. These methods help determine what information can be efficiently obtained from swarming, multi-sensor laden UAVs with a high degree of certainty (high expectation) that the information will meet the time space needs of the user. The objective is to develop methods for looking at the movement of the swarm around a field of interest and determine the time and place where the most relevant information is likely to be sensed. We integrate these methods into an analytical tool used to gather information metrics about the swarm and allow analysts to draw conclusions about the availability of information.

5.1 Information Quality Expectation

Information quality expectation is introduced in Chapter 1 and defined in Chapter 3 as the "expectation that, based on an analysis of swarm position in time, information highly relevant to a need can be efficiently extracted from the recorded knowledge" Expectation, used in this context, is an expression of confidence of a conclusion drawn from the cumulative results of a set of quantitative and qualitative operations on a set of data.

Through the use of several tools that apply analysis algorithms to positional information, and visualize the movement of the swarm, analysts conclude where the data most relevant to their need is most likely to reside. From there, it is up to the user to decide if the results of the analysis will bring out the needed information and subsequently ask the swarm sensor database for the information.

We can depict and model information quality expectation several ways. Numerous steps are needed to completely study this field. This work looks at the problem from the standpoint of the physical location of the swarm, its members, and the targets. From a geometrical analysis (quantitative) standpoint, expected information is extracted by looking at how the swarm moves about in the area of interest over time. From a qualitative standpoint, witnessing the movement of the swarm around certain targets allows the analyst to see the mission and draw conclusions about the data based on visual queues.

A sensor's proximity to a target impacts the amount and quality of useable data collected. Similarly, the particular phenomenon being sensed carries its own sense-ability sensitivity requirements. Visible sensors see items with less clarity from a distance than from close in. Chemicals disperse over distance and are thus less detectable from afar. Sound diminishes with distance and time, and cannot be trusted as a source of information from anywhere but within relatively close range. This research examines the availability of information that a position-in-time file can provide to analysts looking for ways to extract meaningful data from swarm sensor databases.

Several tools are now available that allow the user to quickly find the combination of time, target and sensor that will help extract the most relevant information for a reconnaissance need. The combination of target-centric and sensor-centric results give the user numerous different ways of analyzing the swarm and how it moves about the area of interest. The quantity of queries needed to retrieve relevant information from the on-board sensor database is reduced by following the flow of information from the position database. The efficiency of the queries is improved and the sensor information returned will be more specific and relevant to the user's needs. The net result is that the information returned from the database will allow for more accurate decisions.

5.2 Swarm Position Analysis Tool

The analysis tool used for this work is designed to be straightforward and easy to use. By implementing the analyzer using primarily Java and MySQL, the portability of the system across platforms is maintained. This analysis tool can be combined with other simulation programs to analyze the effectiveness of their swarming and routing algorithms.

5.3 Swarm Positional Data

The data driver for this research is the positional data from the swarm itself. Without it, the notion of creating an expectation of information fails. While UAV swarms are not yet a practical reality, and the optimal methods for storing positional data are not yet standardized, it is reasonable to assume that this type of information can be generated and maintained for analysis. “Swarm” is used throughout this work to mean UAVs flying in some formation, but the reality is that swarm may represent special forces foot soldiers being tracked by their commanding units or a tank brigade moving on enemy positions. Regardless of the originator of the positional data, it can be analyzed by the tools developed for this work.

5.4 Swarm Visualization

Visualization of the swarm members with each other as well as targets of interest is of significant value to analysts. Because the swarm is unmanned, it is impossible get a first-hand sense of how the agents are working as a team. By using the tools developed here, the interaction of the swarm members is visualized and an analyst is able to judge how the sensor data should look based on the visual cues taken from the software. By allowing the analyst to change the perspective from one UAV to another, or to switch from a sensor-centric view to a target-centric view and replay the segments of the swarm’s flight, anomalies in returned sensor data are better understood.

5.5 Bandwidth Advantages

Broad queries of the sensor database are eliminated by using positional data to determine when and where the most relevant sensor data is collected. Knowing the limits of the sensors on board the UAVs as well as the proximity of the UAVs to targets allows the elimination of large quantities of data which would otherwise be returned from a query to the swarm sensor data store. The actual percentage of data eliminated varies greatly from one information need to another, but as the need becomes more focused, the percentage of low-relevance information increases. The methods developed here allow the user to pinpoint where the needed information is and only ask for that specific information. This reduction of requests and the subsequent reduction in returned information frees up valuable bandwidth across data links for potentially more important uses.

5.6 Summary

The products and techniques developed in this research provide a focused approach for observing swarm movement. The simulator developed for this research analyzes the swarm movement over a field of targets. The times when relevant sensor data was was most likely to have been taken are highlighted through this analysis. Swarm dynamics are such that while the sensors are collecting data continually, the proportion of relevant data to collected data is small. Extracting relevant data from a swarm sensor database can be done two ways. First, the entire data set can be downloaded from the airborne data store and analyzed, using large amounts of bandwidth and time. Second, the optimal times for data collection can be derived through the analysis of positional data and this knowledge can be used to strategically ask for the data from times and sensors that should return relevant information. The latter is demonstrated.

Position data collected from a UAV swarm could easily be stored and never looked at. Sensor data is tagged with the position and time it was collected, and unless the swarm experiences a catastrophic failure, there is no great use for the raw positional data itself. The methods implemented through the Swarm Position Analysis Tool allow analysts to make use of the position data to pare down the amount of sensor data traffic that is passed over data links. With UAV swarms envisioned to be in the range of hundreds of swarm members each, efficient data extraction techniques are needed to ensure only relevant data is sent from the swarm to the analysts.

As the field of swarming UAVs transitions from the theoretical world to the practical world, the possibilities for information acquisition and dissemination are seemingly boundless. Through the careful design of information retrieval methods and practices, the vast amount of gathered information can be filtered to bring forth only that which effectively shortens decision times and more efficiently allows the application of force toward mission accomplishment.

6. Recommendations for Future Work

As is the case with many research projects, time influences how much can be accomplished. In the case of this work, several areas that can be expounded upon to make this simulation more realistic. With improvements in some of these areas, this program could be used as a basis for developing war planning simulators that incorporate UAV swarms into the battle planning.

The following sections outline several areas that can be added along with some suggested direction for those additions.

6.1 Sensor Modeling

To ensure an accurate modeling of the sensor data, it is necessary to define the parameters within which the sensors will gather data. Several types of sensors can be modeled. Examples of a few and their possible modeling techniques are discussed

6.1.1 Electro-Optical Sensors. Perhaps no sensor gives such immediate feedback to the analysts as a video sensor. For this sensor several characteristics need to be defined: field of view (FOV), resolution and straight line distance to target. Field of view refers to the angular divergence of the camera, or the cone emanating from the camera in which objects can be seen. The higher the number for the divergence angle, the larger the "spot" on the ground that is viewed by the camera. Resolution refers to the number of pixels of information that can be collected by the sensor. The higher the number of pixels, the more information can be gathered in each frame. The combination of FOV and resolution defines the size of an object that can be discriminated in the video frame. Table 6.1 shows the effect of either increasing the resolution or FOV for a sensor aboard a UAV at 20,000 ft. The numbers below the resolutions indicate the smallest object (measured in square feet) that can be seen by the sensor.

Table 6.1: Field of View vs. Resolution

FOV (degrees)	Spot Radius (ft)	Area seen (sqft)	Resolution ($pixels^2$) 1600	1280	1024
24	4251	56775228	22.1	34.6	54.1
18	3168	31523533	12.3	19.2	30.0
12	2102	13881944	5.4	8.4	13.2
6	1048	3451448	1.3	2.1	3.2
3	524	861679	0.3	0.52	0.8
1.5	262	215346	0.1	0.13	0.2

Because technical advances have driven the physical size and weight of digital video cameras to extremely small proportions, it is not uncommon for these to be mounted on moving gimbals that allow for scanning over a large angular field from the vehicle. In an effort to bring the modeling of these sensors to within tolerable limits, it could be assumed that the camera is in a fixed location and always looks straight down from the aircraft without any scanning ability.

6.1.2 Chemical Detection Sensors. Chemical detection sensors are used to "sniff" the air to determine if any harmful chemicals exist and in what concentration. They look for minute amounts of chemical particles often measured in the parts per million or parts per billion range. As the amounts of chemicals increase, the reading on the sensor rises. For this experiment, the sensors sample the air once per time slice. As they move, they encounter "clouds" of chemicals emitted by the target and sent downwind. The sensor readings are determined by the distance from the target and the level of chemicals in the air at the point in space. The emission of chemicals is discussed in Section 6.2.2.

6.1.3 Acoustic Sensors. Acoustic sensors are modeled in much the same way as the chemical detection sensors. As the sensor comes in range of the noise field, it begins registering the levels of noise it senses. As the sensor comes closer to the target, the sensor reading correspondingly increases.

6.2 Target Modeling

Central to the accurate modeling of the data collection of a UAV swarm is the modeling of the targets that are to be sought out. Modeling is accomplished using a *target* class. Targets with different characteristics have different values assigned for variables. The following sections describe the methods that could be used to portray targets in the UAV environment.

6.2.1 Electro-Optic Targets.

Electro-optic targets consist of both the visible targets (can be seen by the human eye) and those that emit an IR signature. Both the optically visible and the IR visible targets have a size associated with them. This is necessary for determining if the sensor is capable of seeing the target based on its resolution. The IR targets have, in addition to a size, a randomly assigned IR signature level. This value, along with the size determines whether or not a certain sensor can see the target based on distance from the target. Depending on the target, IR sensors can often pick out targets better than a sensor operating in the visible spectrum because of the contrast between the target and the background. This is especially true at night.

6.2.2 Chemical Targets.

Chemical targets are modeled using two parameters, wind direction and chemical level. Upon the start of a simulation, a wind direction is assigned to the area of interest. All chemical targets inherit that value. Each individual chemical target is randomly assigned a chemical level. During the simulation, all chemicals "drift" in the direction of the wind. Based on the level of chemical emitted from the target, the area of detectable dispersion is determined to be a triangle of some proportions with the chemicals dispersing in symmetric angles from the target in the direction of the wind. As the distance from the target increases, the level of emissions degrades in a pre-determined fashion to a threshold where it will be determined that no detectable chemicals exist.

6.2.3 Acoustic Targets. Much like the chemical targets, the acoustic target signal strength degrades as the distance from the target increases. The difference is that the sound waves emanate omnidirectionally from the target, meaning that any acoustic sensor that comes within a distance less than the total degradation distance registers as having heard the target. When the target is created, a random number is assigned that indicates the amplitude of the acoustic signal the target is transmitting. As the distance from the target increases, the level of emissions degrades in a linear fashion to a threshold where it is determined that no detectable acoustic signal exist.

6.3 Sensor Database

The sensor database used in this work represents the foundation for future work. Time restraints kept the implementation of sensors and sensor use to simple boolean values that indicated the presence of sensors on board the UAV. Future work could focus on modeling the data collection and storage of simulated sensor data. The implementation of this would give the researcher the ability to then query the data base for sensed values at the times determined to be of the highest reliability by the Target and Sensor centric methods developed in this project.

6.3.1 Data Collection. As the UAVs move through their track, the simulator constantly monitors their position and their sensor capabilities and compares them with the targets. As long as there are no targets in “view” of the sensors, the database is populated with a “0” in the data field corresponding to the time/sensor combination. As a target comes into ”view” of the sensor, readings are increased to some pre-determined maximal value that indicates the best possible reading from the sensor for that particular target.

During the data runs, the simulator creates the sensor data file and populates the *UAVS, Time* and *Targets* tables prior to the data run. While the simulation is in progress, a new table, the *SensorData* table is populated. This table is the main information table in the database and is by far the biggest. For a 10,000 time-step data run with 50 UAVs, the table is populated with 500,000 entries. Every UAV at every time step is accounted for in this table. If the UAV does not have a certain type of sensor, a “-1” is entered in the table. If the UAV has the sensor and it is not seeing a target, a “0” is entered. If there is a target within the field of view, the sensor reading is recorded in the table. Whenever a sensor reading is above 0, an entry is made in the *TargetAcquisition* table. This table is to be used for comparison of data sizes as well as multiple runs of simulations to extract various metrics.

6.4 Terrain Modeling

The terrain in the today’s current world threat areas ranges from wide open desert to rugged mountain terrain with bodies of water in between. All of this plays an important role in the ability of different types of sensors to take measurements. Video sensors can’t see if there is insufficient ambient light to illuminate the targets, or if the targets are in the shadows. Video and infrared sensors will fail to see targets behind a hill or obscured by vegetation. Readings from acoustic sensors can be thrown off by noise reflecting off mountains and seemingly coming from false directions, and chemical sensors can be fooled by a sand storm.

Accurate digital terrain and elevation data (DTED)is widely available and can be integrated into this simulation. By placing the targets on the ground in varying terrain environments, the simulation can produce more accurate results with regard to data modeling.

Because the position units in the track files are simple X-Y coordinates, it is possible to transform these into actual latitude and longitude points. This is necessary prior to using DTED as the DTED is set up using the lat-lon coordinate system. The scale of these can be changed by simply moving the decimal point in the X-Y system. This allows the user to increase or decrease the area of coverage.

By integrating altitude data from both the targets and the UAVs, look angles can be calculated from the UAV to the target. By knowing these along with the terrain in between, accurate data collection models are possible.

6.5 Time of Day Modeling

The time of day that the mission takes place has a large impact on the data collected by the sensors. The global threat in today's world resting largely in the middle east. Because of this, most missions flown today take place over arid environments prone to large temperature fluctuations throughout any given 24 hour period.

In this simulation, the time of day for the start of the simulation could be entered along with a time slice size. Each time slice represents a user defined amount of time. For a 1000 time slice simulation, a slice size of one second represents a total simulation time of 1000 seconds or 16 minutes and 20 seconds. If that slice is expanded to fifteen seconds, the simulation then runs for a "period" of 250 minutes, or four hours and ten minutes. In the first scenario, the targets will not change much over the simulation time due to environmental changes. In the second scenario, however, the visibility of the targets could change significantly due to such changes as the rising or setting sun. At these times, visual sensors such as video or infrared detectors can be significantly effected by dynamic light and thermal conditions. For this simulation, the sunrise and sunset times could be set using the Simulation Setup user interface. The time step is also entered. Once the simulation passes a threshold for sunrise or sunset, the sensitivity of the video sensor is set to a value such that it either sees the targets (sunrise)or has no ability to see and thus will not collect any data (sunset).

6.6 Integration of Other Research

This simulation program was written to be able to be used by more than just UAV simulators. The input to the program is simply a position in time data file. Any tracking program or simulator that can output this type of data can feed the simulator and can be analyzed. The visualization tools allow the user to replay history and extract numbers that give information about what was happening where.

Bin 2000, Berridge [5] focused his research on extracting mission semantics from unmanned aerial vehicle telemetry and flight plans. This type of research works hand in hand with the tools developed here. By paring down the field of potential data sources to a few based on the geometric analysis of movement, the scene is set for research like Berridge's to take the ball from there and fine exact moments where the relevant data is actually in place.

With the future of UAV research at AFIT seemingly strong, this tool allows researchers to visualize their algorithms in an easy to use way. Two and three dimensional views of the swarm will allow instant feedback on whether designed algorithms will be feasible or not.

Appendix A. SQL Code

This appendix details the Structured Query Language (SQL) statements used in this research. Some of these are simple statements that merely insert and retrieve data, and some are more complex, calculating parameters and updating tables.

A.1 UAV IDs

The following SQL statement extracts the UAV identifications from the track file table. The result is used to set up the UAV objects.

```
SELECT distinct UAVID
FROM uavpositions
```

A.2 Area Of Interest

This SQL statement finds the minimum and maximum latitude and longitude points for the swarm. This is used to set up the bounds of the simulated area of interest. `SELECT max(xpos), min(xpos), max(ypos), min(ypos)`

```
FROM uavpositions
```

A.3 Centers Of Mass

The following SQL statement generates the numbers for the centers of mass of the whole swarm and the sub swarms broken out by sensor family for each time step. This statement was developed by Mr. Joseph Shapiro, AFIT GCS-05M

```
INSERT INTO CentersOfMass (Time, swcomxpos, swcomypos, videocomxpos, videocomypos,
ircomxpos, ircomypos, chemcomxpos, chemcomypos, acousticcomxpos, acousticcomypos)
SELECT poso.time as time,
(SELECT avg(pos.xpos) FROM uavs uv,uavpositionspos
WHERE uv.uavid = pos.uavid and pos.time = poso.time) as SWCOMXpos,
(SELECT avg(pos.ypos) FROM uavs uv, uavpositions pos
WHERE uv.uavid = pos.uavid and pos.time = poso.time) as SWCOMYpos,
(SELECT avg(pos.xpos) FROM uavs uv, uavpositions pos
WHERE uv.uavid = pos.uavid and uv.isvisible = 'true' and pos.time = poso.time) as VideoCOMXpos,
(SELECT avg(pos.ypos) FROM uavs uv, uavpositions pos
WHERE uv.uavid = pos.uavid and uv.isvisible = 'true' and pos.time = poso.time) as VideoCOMYpos,
(SELECT avg(pos.xpos) FROM uavs uv, uavpositions pos
WHERE uv.uavid = pos.uavid and uv.isir = 'true' and pos.time = poso.time) as IRCOMXpos,
(SELECT avg(pos.ypos) FROM uavs uv,uavpositions pos
WHERE uv.uavid = pos.uavid and uv.isir = 'true' and pos.time = poso.time) as IRCOMYpos,
(SELECT avg(pos.xpos) FROM uavs uv, uavpositions pos
WHERE uv.uavid = pos.uavid and uv.ischem= 'true' and pos.time = poso.time) as ChemCOMXpos,
(SELECT avg(pos.ypos) FROM uavs uv,uavpositions pos
WHERE uv.uavid = pos.uavid and uv.ischem= 'true' and pos.time = poso.time) as ChemCOMYpos,
(SELECT avg(pos.xpos) FROM uavs uv, uavpositions pos
WHERE uv.uavid = pos.uavid and uv.isacoustic= 'true' and pos.time = poso.time) as AcousticCOMXpos,
(SELECT avg(pos.ypos) FROM uavs uv, uavpositions pos
WHERE uv.uavid = pos.uavid and uv.isacoustic= 'true' and pos.time = poso.time) as AcousticCOMYpos
FROM uavpositions poso GROUP BY poso.time ;
```

Appendix B. Vendor Software and Java Code

This appendix outlines the vendor software and sections of the Java programming code used in key areas of the software written for this work. These code snippets represent portions of the program that computed critical metrics or determines important set up parameters for the analysis tool.

B.1 Vendor Software

Table B.1 lists the software, vendor and the applications used to create the simulator. All of the components are freely available from the internet.

Table B.1: Software Used in Development

Product	Vendor	Used For
Java Development Kit 1.5	Sun Microsystems	Overall System Development
NetBeans 4.0	Sun Microsystems	Integrated Development Environment for Java code development
Eclipse 3.0.1	IBM	Integrated Development Environment for Java and database code development
MySQL 4.1.7 Relational Database	MySQL	Database Implementation
Audit Trail Viewer 2.1	Air Force Research Laboratories	Visualization
ConvexHull.java	Starlink	Convex Hull calculations

B.2 Java Code

B.2.1 UAV Setup. The following code creates the UAVs and initializes the sensors:

```
private void readDataFile(String inString) {
        String directory = "C:\\\\Documents and Settings\\\\Patrick\\\\"+
        "My Documents\\\\AFIT\\\\THESIS\\\\BaldwinThesis\\\\trackfiles\\\\";

        // Load track file into database
        String query = "LOAD DATA LOCAL INFILE '" + directory + inString +
        "' INTO TABLE uavpositions FIELDS TERMINATED BY ',' ENCLOSED BY '\"'";
        dw.createThingsInDatabase(query);

        // Retrieve distinct uavs from database and generate UAV objects from
        // them
        rs = dw.getRequestedData("SELECT distinct uavid from uavpositions");
        System.out.println("Selected UAV IDs");
        try {
            while (rs.next()) {
                String UAVID = rs.getString("uavid");
                UAVs.put(new Integer(uavIdKey), new UAV(UAVID));
                uavIdKey++;
            }
        } catch (SQLException e) {
            System.out.println("ERROR GENERATING UAVS: " + e);
            e.printStackTrace();
        }

        System.out.println("Setting Up Sensors");
        setUpUAVSensors(vidSens, "Video");
        setUpUAVSensors(irSens, "IR");
        setUpUAVSensors(chemSens, "Chemical");
        setUpUAVSensors(acousticSens, "Acoustic");
```

```
} // End readDataFile


/**
 * Randomly assigns sensors to the UAVs. Some UAVs will have multiple
 * sensors assigned to them. Some may not have any sensors assigned to them
 *
 * @param numType:
 *            The number of UAVs that will be assigned with a given sensor
 *            type
 * @param type:
 *            The type of sensor being assigned to the UAV
 */
private void setUpUAVSensors(int numType, String type) {
    System.out.println("Setting up " + type + " sensors");
    Double holder = new Double(((double) uavIdKey) *
                            (((double) numType) / 100.0));
    int numSensorsToAssign = holder.intValue();
    Vector sensorAssignments = new Vector();
    for (int i = 1; i < numSensorsToAssign + 1; i++) {
        Double tempDbl = new Double(Math.random() * uavIdKey);
        int temp = tempDbl.intValue();
        if (temp == 0)
            temp += 1; // protects from looking for a "0" key in TreeMap
        boolean isSelected = false;
        Iterator check = sensorAssignments.iterator();
        while (check.hasNext()) {
            if (temp == ((Integer) check.next()).intValue())
```

```
                isSelected = true;
            }

            if (!isSelected) {
                sensorAssignments.add(new Integer(temp));
                if (type == "Video")
                    ((UAV) UAVs.get(new Integer(temp))).setIsVisible(true);
                else if (type == "IR")
                    ((UAV) UAVs.get(new Integer(temp))).setIsIR(true);
                else if (type == "Chemical")
                    ((UAV) UAVs.get(new Integer(temp))).setIsChemical(true);
                else if (type == "Acoustic")
                    ((UAV) UAVs.get(new Integer(temp))).setIsAcoustic(true);
            } else {
                i--;
            }

        }
        System.out.println("Set up " + type + " sensors");
    }
```

B.2.2 Area Of Interest Determination. The following code was used to determine the bounds of the area of interest for the simulation:

```
try {
            rs = dw.getRequestedData("SELECT min(xpos) as minlat,"+
                                    " max(xpos) as maxlat,"+
                                    " min(ypos) as minlon,"+
                                " max(ypos) as maxlon "+
                                " FROM uavpositions");
            while (rs.next())
            {
                minlat = rs.getDouble("minlat");
                maxlat = rs.getDouble("maxlat");
                minlon = rs.getDouble("minlon");
                maxlon = rs.getDouble("maxlon");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        // Sets up a buffer around the boundaries of the swarm movement
        double latBuff = (maxlat - minlat)*.05;
        double lonBuff = (maxlon - minlon)*.05;
        minlat -= latBuff;
        maxlat += latBuff;
        minlon -= lonBuff;
        maxlon += lonBuff;
```

B.2.3 Target Placement. The following code was used to assign the placement of the targets and to load the targets in to a tree map container:

```
TreeMap Targets = new TreeMap(); double lat = 0.0;
        double lon = 0.0;
        double LATRANGE = (maxlat - minlat);
        double LONRANGE = (maxlon - minlon);
        for (int i = 1; i < numTargets + 1; i++) {
            lat = (Math.random() * LATRANGE) + minlat;
            lon = (Math.random() * LONRANGE) + minlon;
            String name = "TGT" + Integer.toString(i);
            Targets.put(new Integer(i), new Target(name, new Point2D.Double(
                    lat, lon)));
        }
```

B.2.4 Target Assignments. The following code was used to assign the phenomena to the targets:

```
private void setUpTargetTypes(int numType, String type) {
        Double holder = new Double(((double) numTargets)
                * (((double) numType) / 100.0));
        int numTargetsToAssign = holder.intValue();
        Vector tgtEmission = new Vector();
        for (int i = 1; i < numTargetsToAssign + 1; i++) {
            Double tempDbl = new Double(Math.random() * numTargets);
            int temp = tempDbl.intValue();
            if (temp == 0)
                temp += 1;
            boolean isSelected = false;
            Iterator check = tgtEmission.iterator();
            while (check.hasNext()) {
                if (temp == ((Integer) check.next()).intValue())
                    isSelected = true;
            }
            if (!isSelected) {
                tgtEmission.add(new Integer(temp));
                if (type == "Video")
                    ((Target) Targets.get(new Integer(temp)))
```

```
                            .setIsVisible(true);
                else if (type == "IR")
                    ((Target) Targets.get(new Integer(temp))).setIsIR(true);
                else if (type == "Chemical")
                    ((Target) Targets.get(new Integer(temp)))
                            .setIsChemical(true);
                else if (type == "Acoustic")
                    ((Target) Targets.get(new Integer(temp)))
                            .setIsAcoustic(true);
            } else
                i--;
        }
    }
}
```

B.2.5 Convex Hulls. The following code calculated and stored the convex hulls of the swarms and sub swarms:

```
 public void convexHullOps() {
        int minTime = 0;
        int maxTime = 0;
        int numUavs = 0;
        double area = 0.0;
        double density = 0.0;
        ResultSet rs = null;
        String query = "";
        String query1 = "";
        String family = "";
        Vector SWPoints = new Vector();
```

```
TreeMap SWArea = new TreeMap();
TreeMap SWDensity = new TreeMap();
TreeMap VDArea = new TreeMap();
TreeMap VDDensity = new TreeMap();
TreeMap IRArea = new TreeMap();
TreeMap IRDensity = new TreeMap();
TreeMap CHArea = new TreeMap();
TreeMap CHDensity = new TreeMap();
TreeMap ACArea = new TreeMap();
TreeMap ACDensity = new TreeMap();

for (int k = 0; k < 5; k++) {

    //This breaks out the subswarms based on sensor family
    switch (k) {
        case 0:
            query = "SELECT count(distinct UAVid) as uavNum,
            min(time) as mint,
            max(time) as maxt from uavpositions";
            query1 = "SELECT xpos, ypos
            FROM uavpositions where time = " ;
            family = "Swarm";

            break;
        case 1:
            query = "SELECT count(distinct pos.UAVid) as uavNum,
            min(pos.time) as mint,
            max(pos.time) as maxt
        FROM uavs uv, uavpositions pos
        WHERE uv.uavid = pos.uavid and  uv.isvisible = 'true' ";
        query1 = "SELECT pos.xpos as xpos, pos.ypos as ypos
        FROM uavs uv, uavpositions pos where uv.uavid = pos.uavid and
        uv.isvisible = 'true' and pos.time = ";
        family = "Video";
        break;
    case 2:
        query = "SELECT count(distinct pos.UAVid) as uavNum,
        min(pos.time) as mint,
        max(pos.time) as maxt
        FROM uavs uv, uavpositions pos
        WHERE uv.uavid = pos.uavid and  uv.isir = 'true' ";
        query1 = "SELECT pos.xpos as xpos, pos.ypos as ypos
        FROM uavs uv, uavpositions pos where uv.uavid = pos.uavid and
        uv.isir = 'true' and pos.time = ";
        family = "IR";
        break;
    case 3:
        query = "SELECT count(distinct pos.UAVid) as uavNum,
        min(pos.time) as mint,
        max(pos.time) as maxt FROM uavs uv, uavpositions pos
        WHERE uv.uavid = pos.uavid and  uv.ischem = 'true' ";
        query1 = "SELECT pos.xpos as xpos, pos.ypos as ypos
        FROM uavs uv, uavpositions pos where uv.uavid = pos.uavid and
        uv.ischem = 'true' and pos.time = ";
        family = "Chem";
        break;
    case 4:
        query = "SELECT count(distinct pos.UAVid) as uavNum,
        min(pos.time) as mint,
        max(pos.time) as maxt FROM uavs uv, uavpositions pos
        WHERE uv.uavid = pos.uavid and  uv.isacoustic = ’true’ ";
        query1 = "SELECT pos.xpos as xpos, pos.ypos as ypos
        FROM uavs uv, uavpositions pos where uv.uavid = pos.uavid and
        uv.isacoustic = ’true’ and pos.time = ";
        family = "Acoustic";
        break;
}
System.out.println("Getting "+ family + "Swarm");
//Find the minimum and maximum time stamps and the
//number of different UAVs in the swarm
rs = getRequestedData(query);
try {
    while (rs.next()) {
        numUavs = rs.getInt("uavNum");
        minTime = rs.getInt("mint");
        maxTime = rs.getInt("maxt");
    }
} catch (Exception e) {
}

//Beginning with the minimum time stamp and ending with the max
// time stamp
//Get the positions of each uav and get the convex hull parameters
for (int i = minTime; i < maxTime + 1; i++) {
```

```
String query2 = query1;
query2 += i;
double x[] = new double[numUavs];
double y[] = new double[numUavs];
rs = getRequestedData(query2);
int insertnum = 0;
try {
    while (rs.next()) {
        x[insertnum] = rs.getDouble("xpos");
        y[insertnum] = rs.getDouble("ypos");
        insertnum++;
    }
} catch (Exception e) {
}
ConvexHull ch = new ConvexHull(x, y);
area = ch.getArea();
density = (double) numUavs / area;
Iterator itr = ch.points();
SWPoints.add(itr);

//Put the areas and densities in the appropriate containers
switch (k) {
    case 0:
        SWArea.put(new Integer(i), new Double(area));
        SWDensity.put(new Integer(i), new Double(density));
        break;
    case 1:
        VDArea.put(new Integer(i), new Double(area));
```

```
                VDDensity.put(new Integer(i), new Double(density));
                break;
            case 2:
                IRArea.put(new Integer(i), new Double(area));
                IRDensity.put(new Integer(i), new Double(density));
                break;
            case 3:
                CHArea.put(new Integer(i), new Double(area));
                CHDensity.put(new Integer(i), new Double(density));
                break;
            case 4:
                ACArea.put(new Integer(i), new Double(area));
                ACDensity.put(new Integer(i), new Double(density));
                break;
        }


    }
    insertCHPoints(family, SWPoints);
    SWPoints.clear();
}

//Insert the densities and areas into the database by stepping through
//the treemaps containing the values and extracting them based on time
String areasIn = "INSERT INTO ch_area_and_density VALUES ";
for (int i = minTime; i < maxTime + 1; i++) {
    double SWA = ((Double)SWArea.get(new Integer(i))).doubleValue();
    double SWD = ((Double)SWDensity.get(new Integer(i))).doubleValue();
```

```
            double VDA = ((Double)VDArea.get(new Integer(i))).doubleValue();
            double VDD = ((Double)VDDensity.get(new Integer(i))).doubleValue();
            double IRA = ((Double)IRArea.get(new Integer(i))).doubleValue();
            double IRD = ((Double)IRDensity.get(new Integer(i))).doubleValue();
            double CHA = ((Double)CHArea.get(new Integer(i))).doubleValue();
            double CHD = ((Double)CHDensity.get(new Integer(i))).doubleValue();
            double ACA = ((Double)ACArea.get(new Integer(i))).doubleValue();
            double ACD = ((Double)ACDensity.get(new Integer(i))).doubleValue();
            areasIn += "("+i+","+SWA+","+SWD+","+VDA+","+VDD+","+IRA+","+IRD+","+
            CHA+","+CHD+","+ACA+","+ACD+"),";

        }

        // Remove ending "," and place into database
        String areasIn1 = areasIn.substring(0,areasIn.length()-1);
        createThingsInDatabase(areasIn1);
    }
```

B.2.6 Target Profiling.

```
private void btnProfileTargetActionPerformed(java.awt.event.ActionEvent evt) {
        int start = ((Integer) cmbStartTime.getSelectedItem()).intValue();
        int end = ((Integer) cmbEndTime.getSelectedItem()).intValue();
        double fogInterval = 0.0;
        if (start < end && cmbTargets.getSelectedItem() != null &&
        !tfRadius.getText().equalsIgnoreCase("")) {
            final Plot myPlot = new Plot();
            if (rbClip.isSelected())
```

```
    myPlot.setTitle("Target " +
    cmbTargets.getSelectedItem().toString() +
    " Profile (Clipped at Radius)");
else if (rbSmooth.isSelected())
    myPlot.setTitle("Target " +
    cmbTargets.getSelectedItem().toString() +
    " Profile (Smooth)");
else if (rbFog.isSelected()){
    myPlot.setTitle("Target " +
    cmbTargets.getSelectedItem().toString() +
    " Profile (Fog Value = " + jsldFogValue.getValue()+ ")");
    fogInterval = getAvgAreaSize() /
    (500.0/ (double)jsldFogValue.getValue());
}

myPlot.setXLabel("Time");
myPlot.setYLabel("Profile Value");
ResultSet UAVs = df.getRequestedData("SELECT UAVID, IsVisible, IsIR,
IsChem, IsAcoustic FROM UAVS");
double rad = new Double(tfRadius.getText()).doubleValue();
double tgtLat = new Double(tfLat.getText()).doubleValue();
double tgtLon = new Double(tfLon.getText()).doubleValue();
String tgtID = cmbTargets.getSelectedItem().toString();
double minlat = tgtLat - rad;
double maxlat = tgtLat + rad;
double minlon = tgtLon - rad;
double maxlon = tgtLon + rad;
```

```
int series = 0;
String family = "";
for (int g = start; g < end + 1; g++) {
    double VidVal = 0.0;
    double IRVal = 0.0;
    double ChemVal = 0.0;
    double AcousticVal = 0.0;
    try {
        double distance = 0.0;
        UAVs.beforeFirst();
        while (UAVs.next()) {
         String UAVID = UAVs.getString("UAVID");
         String IsVisible = UAVs.getString("IsVisible");
         String IsIR = UAVs.getString("IsIR");
         String IsChem = UAVs.getString("IsChem");
         String IsAcoustic = UAVs.getString("IsAcoustic");
         rs = df.getRequestedData("SELECT xpos, ypos
         FROM uavpositions
         WHERE UAVid = '" + UAVID + "' and Time = " + g);
         while (rs.next()) {
          double xpos = rs.getDouble("xpos");
          double ypos = rs.getDouble("ypos");
          distance = Math.sqrt(Math.pow((tgtLat - xpos), 2.0) +
          Math.pow((tgtLon - ypos), 2.0));
          if (rbClip.isSelected()) {
           if (distance <= rad) {
              double value = rad / distance;
              if (IsVisible.equalsIgnoreCase("true"))
                 VidVal += value;
              if (IsIR.equalsIgnoreCase("true"))
                 IRVal += value;
              if (IsChem.equalsIgnoreCase("true"))
                 ChemVal += value;
              if (IsAcoustic.equalsIgnoreCase("true"))
                 AcousticVal += value;
              }
         } else if(rbSmooth.isSelected()){
              double value = rad / distance;
              if (IsVisible.equalsIgnoreCase("true"))
                  VidVal += value;
              if (IsIR.equalsIgnoreCase("true"))
                  IRVal += value;
              if (IsChem.equalsIgnoreCase("true"))
                  ChemVal += value;
              if (IsAcoustic.equalsIgnoreCase("true"))
                  AcousticVal += value;
          }else if (rbFog.isSelected()){
                  double exp = Math.floor(distance / fogInterval);
                   double value = 1 / Math.pow(2.0, exp);
              if (IsVisible.equalsIgnoreCase("true"))
                   VidVal += value;
              if (IsIR.equalsIgnoreCase("true"))
                   IRVal += value;
              if (IsChem.equalsIgnoreCase("true"))
                   ChemVal += value;
              if (IsAcoustic.equalsIgnoreCase("true"))
```

```
                       AcousticVal += value;

                    }

                }
            }
        } catch (Exception e) {
            System.out.println("Error retrieving query results.
            Error was: " + e);
        }
        myPlot.addPoint(0, g, VidVal, true);
        myPlot.addPoint(1, g, IRVal, true);
        myPlot.addPoint(2, g, ChemVal, true);
        myPlot.addPoint(3, g, AcousticVal, true);
    }

    myPlot.addLegend(0, "Video");
    myPlot.addLegend(1, "Infrared");
    myPlot.addLegend(2, "Chemical");
    myPlot.addLegend(3, "Acoustic");
    PlotFrame pf = new PlotFrame("Profile for " + tgtID, myPlot);
    pf.setVisible(true);

} else {
    JOptionPane jop = new JOptionPane();
    jop.showMessageDialog(this, "Check Target, Radius or Time",
    "Profiler Error", JOptionPane.ERROR_MESSAGE);
}
```

```
    }
```

B.2.7 UAV Proximity Query. The following code selects the UAVs that came within the selected distance to the target, calculates the distance and bearing and shows the UAVid, Time, position, distance and bearing to the user:

```
void btnRunQueryActionPerformed(java.awt.event.ActionEvent evt) {
       String tgt = ((String) cmbTargets.getSelectedItem()).toString();
       double lat = new Double(tfLat.getText()).doubleValue();
       double lon = new Double(tfLon.getText()).doubleValue();
       double rad = new Double(tfRadius.getText()).doubleValue();
       double tgtLat = new Double(tfLat.getText()).doubleValue();
       double tgtLon = new Double(tfLon.getText()).doubleValue();
       double minlat = lat - rad;
       double maxlat = lat + rad;
       double minlon = lon - rad;
       double maxlon = lon + rad;
       double dist = 0.0;
       double bearing = 0.0;

       String query = "SELECT pos.UAVId as UAVid, pos.Time as Time,
       pos.xpos as xpos, pos.ypos as ypos " +
       "FROM uavs uv, uavpositions pos " +
       "WHERE pos.xpos > " + minlat + " and pos.xpos < " +
       maxlat + "and pos.ypos > " + minlon +
       " and pos.ypos < " + maxlon;
       if (chkbxAll.isSelected()) {

       }
if (chkbxVideo.isSelected()) {
    query += " and (uv.uavid = pos.uavid and
    (uv.isvisible = 'true'))";
}
if (chkbxInfrared.isSelected()) {
    if (chkbxVideo.isSelected()) {
        query = query.substring(0, query.length() - 2);
        query += " or uv.isir = 'true'))";
    } else
        query += " and (uv.uavid = pos.uavid and
        (uv.isir = 'true'))";

}
if (chkbxChemical.isSelected()) {
    if (chkbxVideo.isSelected() || chkbxInfrared.isSelected()) {
        query = query.substring(0, query.length() - 2);
        query += " or uv.ischem = 'true'))";
    } else
        query += " and (uv.uavid = pos.uavid and
                        (uv.ischem = 'true'))";
}
if (chkbxAcoustic.isSelected()) {
    if (chkbxVideo.isSelected() || chkbxInfrared.isSelected() ||
    chkbxChemical.isSelected()) {
        query = query.substring(0, query.length() - 2);
        query += " or uv.isacoustic = 'true'))";
    } else
        query += " and (uv.uavid = pos.uavid and
                (uv.isacoustic = 'true'))";
}

String out[] = { "UAV", "Time", "Lat", "Lon", "Distance", "Bearing" };
;
rs = df.getRequestedData(query);

try {
    rs.last();
    String data1[][] = new String[rs.getRow()][6];
    rs.beforeFirst();
    int x = 0;
    while (rs.next()) {
        String uavId = rs.getString("UAVId");
        String timeStamp = rs.getString("Time");
        double xpos = rs.getDouble("xpos");
        double ypos = rs.getDouble("ypos");
        dist = Math.sqrt(Math.pow((tgtLat - xpos), 2.0) +
                        Math.pow((tgtLon - ypos), 2.0));
        if (dist <= rad) {
            bearing = Math.toDegrees(Math.atan2((tgtLat - xpos),
                                    (tgtLon - ypos))) + 180;
            data1[x][0] = uavId;
            data1[x][1] = timeStamp;
            data1[x][2] = DF.format(xpos);
            data1[x][3] = DF.format(ypos);
            data1[x][4] = DF.format(dist);
            data1[x][5] = DF1.format(bearing);
```

```
                x++;
            }
        }
        setupTable(data1, out);
    } catch (Exception e) {
        System.out.println("Error retrieving query results.  Error was: " + e);
    }

    //tblQueryResults

}
```

Bibliography

1. Northrop Grumman Global Hawk
URL:http://www.is.northropgrumman.com/products/usafproducts.

2. AFRL/IF, Rome, NY
URL: http://www.rl.af.mil/tech/programs/jview.

3. University of California Berkely
URL: http://ptolemy.eecs.berkeley.edu/ptolemyII.

4. Barber, C. Bradford, David P. Dobkin, and H Huhdanpaa. *The Quickhull Algorithm for Convex Hulls*. Technical report, University of Minnesota 1995.

5. Berridge, Walter. *Extracting Mission Semantics From Unmanned Aerial Vehicle Telemetry and Flight Plans*. Master's thesis, Air Force Institute of Technology March 2000.

6. Bonnett, Philippe, Johannes Gehrke, and Praveen Seshadri. *Towards Sensor Database Systems*. Technical report, Computer Science Department Cornell University Ithaca, NY 2000.

7. Kadrovach, Anthony. *A Communications Modeling System forSwarm-based Sensors*. Ph.D. thesis, AirForce Institute of Technology 2003.

8. Krock, L. Spies that Fly: A Timeline of UAVs Nov 2002.

9. Lotspiech, James T. *Distributed Control Of A Swarm Of Autonomous Unmanned Aerial Vehicles*. Master's thesis, Air Force Institute Of Technology 2003.

10. Pfoser, Dieter, Christian Jensen, and Yannis Theodoridis. "Novel Approaches to the Indexing of Moving Object Trajectories". *Proceedings of the 26th International Conference on Very Large Databases, Cairo, Egypt*, 395 – 406 2000.

11. Reynolds, Craig W. Flocks, Herds and Schools: A Distributed Behavioral Model. *Computer Graphics*, volume 21, 25–34 1987.

12. Staples, Edward. A New Electronic Nose. *Sensor Magazine* May, 1999.

13. Webster, M. *Webster's II New Riverside University Dictionary*. Riverside 1994.

Vita

1Lt Patrick Baldwin was born in Middlebury, Vt in August of 1966 and graduated from Middlebury Union High School in June of 1984. He enlisted in the Air Force and came on Active Duty in August of 1985 as an instrumentation and telemetry systems mechanic.

Following Basic and Technical training, he was stationed at Kirtland AFB, NM in May, 1986. He worked in the Nuclear Effects division of the Air Force Weapons Laboratory studying the effects of Electromagnetic Pulse on several weapon systems. In May, 1990 he became a member of the Flight Test Branch, also at the Weapons Lab. In this position he designed mechanical systems flown during numerous experiments on board the C-135E ARGUS airborne test bed.

In December of 1996, then SSgt Baldwin received an assignment to the United States Air Force Academy in Colorado Springs, CO. There he worked as a lab technician in the Dept of Aeronautics as well as taught courses in Mathematics and Computer Aided Design.

January of 2001, TSgt Baldwin attended Officer Training School at Maxwell AFB, receiving his commission in April of that year. After OTS, Lt Baldwin was assigned to the 93rd Air Control Wing which would later be named the 116th Air Control Wing at Robins AFB, GA.

In August of 2003 Lt Baldwin began his assignment at AFIT. Following AFIT, he will be assigned to USSTRATCOM at Offutt AFB, NE

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 074-0188*

1. REPORT DATE *(DD-MM-YYYY)* 03-21-2005	**2. REPORT TYPE** **Master's Thesis**	**3. DATES COVERED** *(From – To)* Aug 2003 – Mar 2005

4. TITLE AND SUBTITLE **Modeling Information Quality Expectation in Unmanned Aerial Vehicle Swarm Sensor Databases**	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Baldwin, Patrick D., 1Lt, USAF	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology, Graduate School of Engineering and Management (AFIT/EN), 2950 Hobson Way, WPAFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT/GCS/ENG/05-01
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/IF, Attn: Mr. Douglas Holzhauer, 26 Electronic Parkway, Rome, NY 13441-4514 DSN: 587-4920	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Swarming Unmanned Aerial Vehicles (UAVs) are the future of Intelligence, Surveillance and Reconnaissance (ISR). Swarms of hundreds of these vehicles, each equipped with multiple sensors, will one day fill the skies over hostile areas. As the sensors collect hundreds of gigabytes of data, telemetry data links will be unable to transmit the complete data picture to the ground in real time. The collected data will be stored on board the UAVs and selectively downloaded through queries issued from analysts on the ground.

Analysts expect to find relevant sensor data within the collection of acquired sensor data. This expectation is not a quantified value, rather a confidence that this relevant data exists. An expectation of the likely quality of the available sensor information is determined by the user through the use of the methods and tools developed in this thesis.

This work develops swarm coverage analysis models using position in time data from the swarm. With these models, a geometric analysis of the swarm is conducted that shows analysts when and where the swarm likely collected sensor data most relevant to a need. Convex hulls are used to calculate areas of coverage as well as swarm and sensor densities. Target profiling algorithms are developed that show target coverage over time from the swarm for each sensor type. Target-centric and sensor-centric analyses allow analysts to quickly determine where individual swarm agents were relative to a target at any point during the mission. Finally a series of visualizations of the swarm and targets are created that allow the analyst to view swarm activity from the perspective of individual swarm members or targets.

15. SUBJECT TERMS
Unmanned Aerial Reconnaissance, Mission Area Analysis, Computerized Simulation, Information Processing, databases

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Michael L. Talbert, LtCol, USAF (ENG)
REPORT U	**ABSTRACT** U	**c. THIS PAGE** U	UU	122	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-2024; e-mail: Michael.Talbert@afit.edu

Standard Form 298 (Rev: 8-98)
Prescribed by ANSI Std. Z39-18